Des stratégies évolutionnaires globalement convergentes avec une application en imagerie sismique pour la géophysique

Youssef Diouane

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Institut National Polytechnique de Toulouse (INP Toulouse)*

Présentée et soutenue le *17/10/2014* **par :**

Youssef DIOUANE

Globally convergent evolution strategies with application to an Earth imaging problem in geophysics

---

JURY

Henri Calandra	Total, USA	President of jury
Serge Gratton	INPT, France	PhD advisor
Luis Nunes Vicente	University of Coimbra, Portugal	PhD co-advisor
Stefano Lucidi	University of Rome, Italy	Referee
Thomas Baeck	Leiden University, Netherlands	Referee
Xavier Vasseur	CERFACS, France	Member of jury

École doctorale et spécialité :
MITT : Domaine Mathématiques : Mathématiques appliquées
Unité de Recherche :
Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique (CERFACS)
Directeurs de Thèse :
Serge GRATTON et *Luis Nunes VICENTE*
Rapporteurs :
Thomas BAECK et *Stefano LUCIDI*

Résumé

Au cours des dernières années, s'est développé un intérêt tout particulier pour l'optimisation sans dérivée. Ce domaine de recherche se divise en deux catégories: une déterministe et l'autre stochastique. Bien qu'il s'agisse du même domaine, peu de liens ont déjà été établis entre ces deux branches. Cette thèse a pour objectif de combler cette lacune, en montrant comment les techniques issues de l'optimisation déterministe peuvent améliorer la performance des stratégies évolutionnaires, qui font partie des meilleures méthodes en optimisation stochastique.

Sous certaines hypothèses, les modifications réalisées assurent une forme de convergence globale, c'est-à-dire une convergence vers un point stationnaire de premier ordre indépendamment du point de départ choisi. On propose ensuite d'adapter notre algorithme afin qu'il puisse traiter des problèmes avec des contraintes générales. On montrera également comment améliorer les performances numériques des stratégies évolutionnaires en incorporant un pas de recherche au début de chaque itération, dans laquelle on construira alors un modèle quadratique utilisant les points où la fonction coût a déjà été évaluée.

Grâce aux récents progrès techniques dans le domaine du calcul parallèle, et à la nature parallélisable des stratégies évolutionnaires, on propose d'appliquer notre algorithme pour résoudre un problème inverse d'imagerie sismique. Les résultats obtenus ont permis d'améliorer la résolution de ce problème.

Mots-clés: Optimisation numérique, stratégies évolutionnaires, convergence globale, décroissance suffisante, problèmes inverses, imagerie du sous-sol, inversion des formes d'ondes acoustiques, calcul parallèle (HPC).

Abstract

In recent years, there has been significant and growing interest in Derivative-Free Optimization (DFO). This field can be divided into two categories: deterministic and stochastic. Despite addressing the same problem domain, only few interactions between the two DFO categories were established in the existing literature. In this thesis, we attempt to bridge this gap by showing how ideas from deterministic DFO can improve the efficiency and the rigorousness of one of the most successful class of stochastic algorithms, known as Evolution Strategies (ES's).

We propose to equip a class of ES's with known techniques from deterministic DFO. The modified ES's achieve rigorously a form of global convergence under reasonable assumptions. By global convergence, we mean convergence to first-order stationary points independently of the starting point. The modified ES's are extended to handle general constrained optimization problems. Furthermore, we show how to significantly improve the numerical performance of ES's by incorporating a search step at the beginning of each iteration. In this step, we build a quadratic model using the points where the objective function has been previously evaluated.

Motivated by the recent growth of high performance computing resources and the parallel nature of ES's, an application of our modified ES's to Earth imaging geophysics problem is proposed. The obtained results provide a great improvement to known solutions of this problem.

Keywords: Numerical optimization, evolution strategies, global convergence, sufficient decrease,inverse problems, Earth imaging, acoustic full-waveform inversion, high performance computing (HPC).

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible. First and foremost I want to thank my supervisors **Serge Gratton** and **Luis Nunes Vicente**. They have taught me, both consciously and unconsciously, how good research is done. I appreciate their availability for the fruitful discussions which make my PhD experience productive and stimulating. The joy and enthusiasm they have for their research was contagious and motivational for me, even during tough times in the PhD pursuit. I am also thankful for the excellent example they have provided as successful researchers.

I am equally grateful to **Henri Calandra** and **Total E&P** for the funding on my PhD, without which this great experience would have not been possible, and for the very challenging geophysical application that they provided, which justifies all the effort behind my studies. I would like to express my sincere thanks to **Xavier Vasseur** for his daily guidance and advices from which I learned so much, not only for my thesis development but also for my future career. I would like also to thank the referees, **Thomas Baeck** and **Stefano Lucidi**, for their careful and enlightening comments on my research.

I am also grateful to all of the **ALGO team members** at CERFACS for being with me during the past three years. Special thanks in particular to **Selime Gürol** for her help, advices, and encouragement. Many thanks also to **Rafael Lago** for his help during the early stage of my PhD. CERFACS administration would not be that efficient without **Brigitte Yzel** and **Michèle Campassens**. Thanks to them for their permanent support in administrative procedures. They were always available to solve my problems with patience and smile.

My special thanks to my best friend **Elhoucine Bergou**, thanks for all these 6 years spent together. My PhD would not have been the same without you my brother. Very special thanks to **Zineb Ghormi** for her never-ending support, trust, encouragement and understanding. My thanks go to my family and friends: my brothers **Simohamed** and **Ayoub**, my sister **Mariam**, my uncles **Omar** and **Brahim**, **Hamza**, **Abdelhadi**, **Azhar**, **Nabil**, **Bassam**, **Naama**, **M'Barek**, **Daoud**, **Hassan**, ...

Lastly, and most importantly, I wish to thank my parents, **Aicha Ouaziz** and **Hissoune Diouane**. They bore me, raised me, supported me, taught me, and loved me. To them I dedicate this thesis.

Contents

List of Figures

List of Algorithms

List of Tables

To my parents

Chapter 1

Introduction

Nowadays, many practical optimization problems have become often noisy, complex, and not sufficiently explicitly defined to give reliable derivatives. In this thesis, we are interested in optimization problems where derivative information is unavailable or hard to obtain in practice. For instance, optimizing large and complex systems often requires the tuning of many parameters. These parameters are typically set to values that may have some mathematical meaning or that have been found to perform well. The choice of parameters can be done automatically using training data of simulations. In such case, not only it is hard to find the derivatives with respect to the parameters, but also numerical noise and probably non-differentiability issues may appear. As consequence, we have seen a resurgence of interest in Derivative-Free Optimization (DFO) [52].

Derivative-based methods are more adapted to solve large scale optimization problems, typically around 10^6 unknowns or more. These methods can be very efficient when the starting point is accurate enough, but otherwise they suffer from stalled convergence to spurious local minima for non-convex optimization problems. Thus the holy grail of these problems is to warmstart the local optimization procedures by efficiently finding a good initial guess without the need of sophisticated a priori knowledge on the objective function (such as the problem structure, its background, ...). When the number of unknowns included in the optimization can be reduced, it is possible to use a type of DFO methods that are known for their ability to handle hard problems and to find a good initial guess (a starting point leading to a better minimum). Once a starting point is found, derivative-based methods can be applied to refine the problem solution. In the scope of this thesis, we deal with a very large scale seismic imaging inversion problem [167] where we show that some DFO methods can improve the optimization procedure by finding an accurate initial guess from which one can initiate derivative-based methods [126], without any physical knowledge.

DFO methods do not use derivatives information of the objective function or constraints, nor an approximation to the derivatives. Actually, derivatives approximation is often very expensive and can produce misleading results due to the presence of noise. DFO area can be divided into two categories, depending on the methods used to explore the search space. The first category is deterministic DFO algorithms such as model-based methods [49, 130] or direct-search methods [52, 108]. The major drawback of these methods is that they can get easily stuck in a local optimum. The second category is stochastic derivative-free optimization [122, 158], which has been employed to mitigate the defect of the local deterministic methods in the solution of difficult objective functions (e.g. non-smooth and multi-modal). Stochastic derivative-free optimization algorithms aim to be robust when dealing with multi-modal objective functions. Some of these methods are generally inspired by nature, in the same way that random processes are often associated with natural systems (e.g. mutations of genetic information, annealing process of metal, molecular dynamics, or swarm behaviors of birds). Well-known representatives of stochastic methods are simulated annealing [107], particle swarm optimization [103] and evolutionary algorithms [26, 32, 91, 92, 142, 150].

Over the past, stochastic DFO was regarded by the deterministic DFO community as another discipline, and only few interactions between the two DFO categories were established. Meanwhile, stochastic optimization algorithms have been growing rapidly in popularity thanks to some methods that became "industry standard" approaches for solving challenging optimization problems. Such growth led the deterministic DFO community to reconsider their position and it has started recently to include stochastic frameworks in their research topics [27, 73, 115, 129].

Evolution strategies (ES's) are one of these successful stochastic algorithms, seen as a class of evolutionary algorithms that are naturally parallelizable, appropriate for continuous optimization, and that lead to interesting results [23, 37, 145]. Motivated by the industrial demand, we propose in this thesis to equip a class of ES's with known techniques from the deterministic DFO community based on the step size control. The incorporated techniques are inspired by the recent development in direct search methods [18, 52, 53, 74, 166]. Our modifications enhance the performance of the original algorithm particularly for expensive objective function evaluation. The proposed ES's achieve rigorously a form of global convergence under reasonable assumptions. By global convergence, we mean the ability of the algorithm to generate a sequence of points converging to a stationary point regardless the starting point.

The problem under consideration, in this thesis, is of the form

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega, \end{aligned} \tag{1.1}$$

where f is a real-value objective function assumed to be bounded by bellow. The feasible region $\Omega \subset \mathbb{R}^n$ of this problem can be defined by relaxable and/or non-relaxable constraints. Relaxable constraints need only to be satisfied approximately or asymptotically. No violation is allowed when the constraints are non-relaxable (typically, they are bounds or linear constraints).

In Chapter 2, we give a short overview of existing deterministic derivative-free optimization methods and their classification. We present the general framework of model-based methods inside their derivative free context. We emphasize multivariate polynomial interpolation techniques used to build different types of local polynomial interpolation and regression models. We also address (directional) direct-search methods where the sampling is guided by a set of directions with specific features. Key concepts particularly related to the sampling set are also outlined (i.e. positive spanning set, a descent direction and the cosine measure). We end up the chapter by reviewing some of the existing global convergence results regarding the presented direct-search methods.

As our main motivation is to equip a class of ES's with some direct search techniques, Chapter 3 gives an overview of stochastic derivative-free optimization algorithms and in particular ES's, their appearance and history, their basic ideas and principles. We present also some theoretical aspects of ES's,in particular, the main existing global convergence properties of ES algorithms. The chapter closes with a detailed description of CMA-ES [85, 86] regarded as state of the art in stochastic derivative-free optimization.

In Chapter 4, we introduce our first contribution where we show how to modify a large class of ES's for unconstrained optimization in order to rigorously achieve global convergence. The type of ES's under consideration recombines the parent points by means of a weighted sum, around which the offspring points are computed by random generation. The modifications consist essentially in the reduction of the size of the steps whenever a sufficient decrease condition on the function values is not verified. When the latter condition is fulfilled, the step size can be reset to the one maintained by the ES's themselves, as long as it is sufficiently large. We propose ways of imposing sufficient decrease for which global convergence holds under reasonable assumptions (e.g. density of certain limit directions in the unit sphere). Given a limited budget of function evaluations, our numerical experiments have shown that the modified CMA-ES is capable of further progress in function values. Moreover, we have observed that such an improvement

in efficiency comes without significantly weakening the performance of the underlying method in the presence of several local minimizers.

The modified ES is extended to handle general constrained optimization in Chapter 5. Our methodology is built upon the globally convergent evolution strategies previously introduced for unconstrained optimization. Two feasible approaches are encompassed to handle the non-relaxable constraints. In the first approach, the objective function is evaluated directly at the generated sampled points. The feasibility is enforced through an extreme barrier function. The second approach projects the generated sampled points onto the feasible domain before evaluating the objective function. The treatment of relaxable constraints is inspired by the merit function approach [74], where one tries to combine both the objective function and the constraints violation function. In the first numerical experiments, where we consider only unrelaxable constraints, we show that our proposed ES approaches (using the extreme barrier or projection) is competitive with the state of the art solvers for derivative-free bound and linearly constrained optimization. In the second part of our numerical experiments, we test our algorithms based on the merit function approach under the presence of both relaxable and unrelaxable constraints. On the chosen test problems, the merit approach shows promising results compared to the progressive barrier one [19], in particular, for relatively small feasible regions.

The modified ES, proposed in Chapters 4 and 5, evaluates the objective function at a significantly large number of points at each iteration. These evaluations can be used in different ways to speed up the convergence and make ES algorithms more efficient especially for small budgets. The possibility that we explore in Chapter 6 is to use the previously evaluated points to construct surrogate quadratic models for the objective function f. The surrogate models are computed using techniques inspired from model-based methods for deterministic DFO. Our hybrid algorithm has been designed to satisfy the convergence analysis of our globally convergent ES. As expected, our experiments show that incorporating local models improves the performance of our ES in both unconstrained and constrained optimization problems. Regression models are found to be the most efficient quadratic ones within our ES algorithms.

Our target application is the solution of an Earth imaging problem in geophysics. In Chapter 7, without any physical knowledge, we use our globally convergent ES's to find a starting point for an optimization procedure that attempts to drive high-resolution quantitative models of the subsurface using the full information of acoustic waves, known as acoustic full-waveform inversion [167]. The chapter starts with a detailed description of the considered problem. We outline also one possible way to adapt our ES to the acoustic full-waveform inversion problem setting. A subspace approach is used for the parametrization of the problem. Motivated by the recent growth of high performance

computing resources, we propose a highly parallel implementation of our ES adapted to the requirements of the problem. The initial results, obtained in this direction, show that great improvement can be expected in the automation of the full-waveform inversion.

Finally, we draw some conclusions and outline perspectives in Chapter 8.

Chapter 2

Deterministic Derivative-Free Optimization

Deterministic derivative-free optimization (DFO) methods either try to build models of the objective function based on sample function values, i.e. model-based methods [49, 52], or directly exploit a sample set of function evaluations without building an explicit model, i.e. direct-search methods [52, 108]. Motivated by the large number of DFO applications, researchers and practitioners made a significant progress on algorithmic and theoretical aspects of the DFO methods over the past two decades. The most important progress concerns the recent algorithms and proofs of global convergence [17, 49, 52, 108, 149, 166]. By global convergence, we mean the ability of a method to generate a sequence of points converging to a stationary point regardless the starting point. A point is said to be stationary if it satisfies the first order necessary conditions, in the sense that the gradient is equal to zero if the objective function is differentiable or, in the non-smooth case, non-negativity following all directional derivatives of the Clarke generalized derivatives [43]. The book by Conn, Scheinberg and Vicente [52] gives a good review of the state of the art of deterministic DFO with a detailed description of the theoretical background to ensure convergence. The main classes of globally convergent algorithms for derivative-free optimization are:

1. **Trust-region methods** [49, 52, 130], where one minimizes accurate models inside a region of prespecified size. The models are for example built either using interpolation and regression techniques [50] or radial-basis functions [168].

2. **Directional direct-search methods** [52, 108], where sampling is guided by sets of directions with appropriate properties, i.e. sets of directions generating $\mathbb{R}^n$ with non-negative coefficients. Popular algorithms under this class are coordinate

search, pattern search, generalized patern search (GPS) [17], generating set search (GSS) [108], and mesh adaptive direct-search (MADS) [18]. We will often refer to this class of methods simply as direct-search methods.

3. **Simplicial direct-search methods** [52, 128], where optimization is ensured through simplex operations like reflection, expansion, or contraction. A popular example is the Nelder-Mead method [128], which is regarded as the most popular derivative-free method.

4. **Line-search methods** [52, 102], where one tries to optimize the objective function using a simplex gradient. The latter is typically chosen as a gradient of linear interpolation or regression polynomial model. A popular example is the implicit-filtering method of Kelley et al [102].

Only trust-region methods and direct-search methods are going to be explored further in this thesis. The remainder of this chapter is organized as follows: we begin by a short overview about model-based methods, where we present the general framework of trust-region methods including their relationship with regression and quadratic models. The second section is devoted to direct-search methods where we present a class of globally convergent directional direct-search methods. The convergence results on this chapter are announced without proofs. For the proofs we refer the reader to [17, 49, 52, 108, 166] and the references given there.

2.1 Model based methods

Model based methods can be seen as a combination of the trust-region framework with interpolation models of the objective function. Basically in these methods, we construct a local model of the objective function and estimate the new step by minimizing the model inside a region. The model is constructed using points evaluated on a specific point subset. Such point subset must verify some appropriate features so that the models can be well-defined. In this section, we briefly describe the essence of this approach. For more detailed analysis, the reader is referred to [49, 51, 52, 130].

2.1.1 Trust-region framework

The trust-region framework is usually used when derivative information of the objective function is available or at least some estimates to the derivatives can be computed

accurately. A typical trust-region method is as follows: at the k-th iteration, given the current iterate x_k, a model of the form

$$m_k(x_k + s) \;=\; f(x_k) + g_k^\top s + \frac{1}{2} s^\top \; H_k \; s \tag{2.1}$$

(where g_k and H_k correspond to estimates of the gradient and the Hessian, respectively) is minimized in a neighborhood around the current iterate defined by the ball (or the trust-region)

$$B(x_k, \Delta_k) \;=\; \{x \in \mathbb{R}^n | \|x - x_k\| \leq \Delta_k\}. \tag{2.2}$$

centered on x_k and with the radius Δ_k; the norm $\|.\|$ could be an iteration dependent norm, but is usually fixed. Different norm choices can be used depending on the minimization problem, for instance in the unconstrained case, the standard Euclidean norm is more adapted [49, 52]. The infinity norm was shown to be more suited when considering bound constraints [49, 72].

The minimization of the model inside the trust-region leads to a new trial point $x_k + s_k$. To determine if the computed point is successful or not, we evaluate the objective function at the new point $x_k + s_k$ and compare the true reduction in the value of the objective function with the predicted reduction by the model. If the ratio

$$\rho_k \;=\; \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)} \tag{2.3}$$

is larger than a constant $\nu_1 > 0$, the step is then accepted, so the model is updated. The trust-region radius is possibly increased if the success is really significant. When the step is unsuccessful (meaning $\rho_k \leq \nu_1$), the trial point is rejected and the trust-region radius Δ_k is reduced.

The approximation model (2.1) is generally constructed using second-order Taylor series expansion. However, in the derivative-free context, one uses alternative approximation techniques that are not based upon the derivatives of the objective function f. Quadratic interpolation is one of these techniques that can be combined with the trust-region algorithms. For guaranteeing convergence, one needs to impose on the approximation model to be locally accurate enough. The interpolation set as well as the mechanism of maintaining it good enough inside the trust-region are described in the next section. The upcoming results are general interpolation and regression results that have been proven useful while dealing model-based optimization. The subscript k is dropped in the following description for clarity reasons; without loss of information since we make a focus on a given iteration of the trust-region algorithm.

2.1.2 Polynomial interpolation and regression models

In this section, we consider the problem of interpolating known objective function values at a given set Y of interpolation points, $Y = \{y^1, y^2, \ldots, y^p\} \subset \mathbb{R}^n$. We aim to find a model m for which the interpolation condition

$$m(y^j) \;=\; f(y^j) \qquad j = 1, \ldots, p \tag{2.4}$$

holds. We say that a set of points can be interpolated by a polynomial of a certain degree, if for the function f there exists a polynomial m such that (2.4) holds for all the points in the interpolation set Y.

2.1.2.1 Polynomial bases

Let $\mathcal{P}_n^d$ be the space of polynomials of degree $\leq d$ in $\mathbb{R}^n$, and q the dimension of this space. Let $\{\phi_i\}_{i=1}^q$ be a given basis of $\mathcal{P}_n^d$, which is a set of q polynomials of degree $\leq d$. Thus, any polynomial $m \in \mathcal{P}_n^d$ can be written uniquely as

$$m(x) \;=\; \sum_{j=1}^{q} \alpha_j \phi_j(x), \tag{2.5}$$

where $\alpha_\phi = (\alpha_1, \ldots, \alpha_q)^\top \in \mathbb{R}^q$. Different polynomial bases ϕ can be considered, the simplest and the most used polynomial basis is the basis of monomials, known as the natural basis $\bar{\phi}$. Such basis is defined using multi-indices in the following way [52]:

Let a vector $\alpha^i = (\alpha_1^i, \ldots, \alpha_n^i) \in \mathbb{N}^n$ be called a multi-index, and, for any $x \in \mathbb{R}^n$, we define x^{α^i} as

$$x^{\alpha^i} \;=\; \prod_{j=1}^{n} x_j^{\alpha_j^i}.$$

Let also

$$|\alpha^i| \;=\; \sum_{j=1}^{n} \alpha_j^i \quad \text{and} \quad \alpha^i! \;=\; \prod_{j=1}^{n} (\alpha_j^i!).$$

Then the elements of the natural basis are

$$\bar{\phi}_i(x) \;=\; \frac{1}{(\alpha^i)!} x^{\alpha^i}, \qquad i = 0, \ldots, q, \quad |\alpha^i| \;\leq\; d.$$

The natural basis can then be written as follows:

$$\bar{\phi} \;=\; \left\{1, x_1, x_2, \ldots, x_n, \frac{1}{2}x_1^2, x_1x_2, \ldots, \frac{1}{(d-1)!}x_{n-1}^{d-1}x_n, \frac{1}{d!}x_n^d\right\}. \tag{2.6}$$

Consequently, for uni-variate interpolation (i.e. $d = 1$) we have $q = n + 1$, and that $q = \frac{(n+1)(n+2)}{2}$ for a full quadratic interpolation (i.e, $d = 2$).

2.1.2.2 Polynomial interpolation

Using (2.5) and (2.4), the coefficients $\alpha_\phi = (\alpha_1, \ldots, \alpha_q)^\top$ can be found by solving the following equation:

$$\sum_{j=1}^{q} \alpha_j \phi_j(y^i) \;=\; f(y^i) \qquad i = 1, \ldots, p,$$

which can be written as a linear system of the form:

$$M(\phi, Y)\alpha_\phi \;=\; f(Y), \tag{2.7}$$

where the coefficient matrix $M(\phi, Y)$ and right hand side $f(Y)$ of this system are

$$\begin{pmatrix} \phi_1(y^1) & \phi_2(y^1) & \cdots & \phi_q(y^1) \\ \phi_1(y^2) & \phi_2(y^2) & \cdots & \phi_q(y^2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(y^p) & \phi_2(y^p) & \cdots & \phi_q(y^p) \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} f(y^1) \\ f(y^2) \\ \vdots \\ f(y^p) \end{pmatrix}, \text{ respectively.}$$

If the coefficient matrix $M(\phi, Y)$ is square and nonsingular, then the set of points Y is poised with respect to the subspace spanned by ϕ. This means that Y can be interpolated by a unique polynomial from this subspace. When the interpolation set remains poised for small perturbations, the set is called well-poised. If the set Y is poised, then one can solve the linear system and find an interpolation polynomial. However, numerically the coefficient matrix $M(\phi, Y)$ may be ill-conditioned depending on the basis choice $\{\phi_i\}_{i=1}^q$. Thus, in general, the condition number of the matrix $M(\phi, Y)$ is a bad measure of poisedness of Y. However, if one chooses the interpolation basis ϕ as the natural basis of monomials $\bar{\phi}$ and $\hat{Y}$ as a shifted and scaled version of Y such as $\hat{Y} \subset B(0; 1)$, the condition number of $M(\bar{\phi}, \hat{Y})$ can be used to monitor the poisedness of the points set [52, Theorem 3.14].

To incorporate models in the trust-region framework, one has to adapt the model construction to different degrees of freedom (which depend on both the cardinality of the interpolation set and the variable size). For instance, during the first iterations one has

only few points and so can not always construct an interpolation model. When $p = n+1$ points are available, we can build a linear model which is known to be sufficient to make some progress. As far as the number of function evaluations p exceeds $n+1$ but not more than $\frac{1}{2}(n+1)(n+2)$, the coefficient matrix $M(\phi, Y)$ contains more columns than rows, and thus the interpolation polynomials defined by (2.4) are no longer unique for quadratic interpolation. To overcome this problem, one uses under-determined models which have been widely used in many practical DFO implementations (see Section 2.1.2.3). Complete quadratic model can be built once the number of function evaluations is equal to $\frac{1}{2}(n+1)(n+2)$, such models being expected to lead to faster progress. As far as the number of function evaluations p exceeds $\frac{1}{2}(n+1)(n+2)$, regression models can be used (see Section 2.1.2.4). Regression models have been shown to be often better than if we just select the 'best' subset of $\frac{1}{2}(n+1)(n+2)$ points and use the chosen subset to build complete quadratic models [50].

2.1.2.3 Under-determined interpolation models

The interpolation polynomials defined by (2.4) are not unique in this case; different approaches can be used [50, 52]:

Sub-basis models: A simple way to impose the uniqueness of the interpolation polynomials can be ensured by restricting the linear system (2.7) to have a unique solution (by removing $q - p$ columns of $M(\phi, Y)$, their corresponding elements of the solution α_ϕ are set to zero). This approach is in general not very successful, except if we have a priori knowledge on the sparsity structure of the gradient and the Hessian of the objective function. Such information can be exploited by deleting the corresponding columns in the linear system (2.7). Choosing p columns in $M(\phi, Y)$ corresponds to removing polynomials from the basis ϕ to obtain a new one $\tilde{\phi}$. As a consequence, the points set Y has to be well poised with respect to the sub-space generated by $\tilde{\phi}$.

Minimum norm models: The second approach to get a unique polynomial solution for the under-determined system (2.7) is to compute the minimum using l_2-norm of the solution α_ϕ. In this case, the problem to solve is defined as follows :

$$\begin{array}{ll} \min & \frac{1}{2}\|\alpha_\phi\|_2^2 \\ \text{s.t.} & M(\phi, Y)\alpha_\phi \;=\; f(Y) \end{array} . \tag{2.8}$$

Assuming that the coefficient matrix $M(\phi, Y)$ has full row rank, the solution of the problem (2.8) is given by

$$\alpha_\phi \;=\; M(\phi, Y)^\dagger f(Y), \tag{2.9}$$

where $M(\phi, Y)^\dagger$ denotes the Moore-Penrose pseudo-inverse of $M(\phi, Y)$. The latter one can be computed using a QR factorization or a singular value decomposition of the coefficient matrix. The polynomial solution found in (2.9) depends on the choice of the basis ϕ. In practice, it has been observed that it is worthy to consider the minimum l_2-norm when one is working with the natural polynomial basis $\bar{\phi}$ [52, Section 5.1].

Minimum Frobenius norm models: The error bounds on both the objective function and its gradient, for under-determined interpolation models, depend on the norm of the Hessian of the model [52, Theorem 5.4]. Therefore, the motivation of this approach is to build models with a minimum value of the norm of the model Hessian. In the quadratic interpolation case, such minimization is equivalent to minimizing the coefficients α_ϕ related to the quadratic monomials. By splitting the natural basis $\bar{\phi}$ into two parts: a linear $\bar{\phi}_L = \{1, x_1, x_2, \ldots, x_n\}$ and a quadratic $\bar{\phi}_Q = \{\frac{1}{2}x_1^2, x_1x_2, \ldots, \frac{1}{2}x_n^2\}$, the interpolation model can be written as follows:

$$m(x) \;=\; \alpha_L^\top \bar{\phi}_L + \alpha_Q^\top \bar{\phi}_Q,$$

where α_L and α_Q are the solution of the following optimization problem

$$\begin{array}{ll} \min & \frac{1}{2}\|\alpha_Q\|_2^2 \\ \text{s.t.} & M(\bar{\phi}_L, Y)\alpha_L + M(\bar{\phi}_Q, Y)\alpha_Q \;=\; f(Y) \end{array}. \tag{2.10}$$

The corresponding solution $\alpha_{\bar{\phi}} = [\alpha_L, \alpha_Q]$ is called minimum Frobenius norm solution. In fact, due to the choice of the natural basis, solving the problem (2.10) is equivalent to minimizing the Frobenius norm[1] of the Hessian of $m(x)$. The solution of (2.10) exists and is uniquely defined if the following matrix is nonsingular:

$$F(\bar{\phi}, Y) \;=\; \begin{pmatrix} M(\bar{\phi}_Q, Y)M(\bar{\phi}_Q, Y)^\top & M(\bar{\phi}_L, Y) \\ M(\bar{\phi}_L, Y)^\top & 0 \end{pmatrix}.$$

The matrix $F(\bar{\phi}, Y)$ is nonsingular if and only if the coefficient matrix $M(\bar{\phi}_L, Y)$ has full column rank and $M(\bar{\phi}_Q, Y)M(\bar{\phi}_Q, Y)^\top$ is positive definite in the null space of $M(\bar{\phi}_L, Y)$ (the last condition can be ensured if the matrix $M(\bar{\phi}_L, Y)$ has full row rank). In this

[1] The Frobenius matrix norm $\|.\|_F$ is defined for a square matrix A by the $\sqrt{\sum_{1 \le i,j \le n} a_{ij}^2}$.

case, the sample set Y is called poised in the minimum Frobenius norm sense. The coefficients α_L and α_Q are computed by solving first

$$F(\bar{\phi}, Y)\begin{pmatrix}\mu\\ \alpha_L\end{pmatrix} = \begin{pmatrix}f(Y)\\ 0\end{pmatrix}$$

to find α_L and μ the Lagrange multiplier of the problem (2.10), then by computing $\alpha_Q = M(\bar{\phi}_L, Y)^\top \mu$ we complete the model construction.

A variant of the Frobenius norm model is the least Frobenius norm updating of quadratic models [137]. Instead of minimizing the Frobenius norm of the model Hessian, one tries to optimize its change from the current iteration to the previously computed Hessian. The new optimization problem can be formulated as follows:

$$\begin{array}{ll}\min & \frac{1}{2}\|\alpha_Q - \alpha_Q^{old}\|_2^2\\ \text{s.t.} & M(\bar{\phi}_L, Y)\alpha_L + M(\bar{\phi}_Q, Y)\alpha_Q = f(Y)\end{array}. \tag{2.11}$$

This optimization problem is solved through a shifted problem on $\alpha_{dif} = \alpha_Q - \alpha_Q^{old}$ of the type given in (2.10).

Minimum Frobenius norm models and its variant have shown to be the most efficient and successful to build quadratic models and are implemented in many software implementations [52, 138]. The minimization of the change in the Hessian of the model from one iteration to the next works very well in some cases, in particular, when $p = 2n+1$ [137, 138].

Sparse quadratic interpolation: When the structure of the Hessian is sparse, it is possible by using the l_1 norm to recover the sparsity of the constructed model in the under-determined case [28]. In fact, instead of solving (2.10) we construct the following optimization problem

$$\begin{array}{ll}\min & \|\alpha_Q\|_1\\ \text{s.t.} & M(\bar{\phi}_L, Y)\alpha_L + M(\bar{\phi}_Q, Y)\alpha_Q = f(Y)\end{array}. \tag{2.12}$$

where α_Q, α_L, $\bar{\phi}_L$, and $\bar{\phi}_Q$ are defined as in (2.10). Solving (2.12) is doable, since it us a linear program (LP). The sparse quadratic approach is shown to be more advantageous when the Hessian of f has zero entries [28].

2.1.2.4 Regression models

This section is devoted to the case where the number of the points p is more than q, meaning that in the quadratic interpolation case, p exceeds $\frac{1}{2}(n+1)(n+2)$. Under such consideration, the linear system (2.7) is overdetermined and has in general no solution. The regression models key idea is to find the best solution that minimizes the gap between the $M(\phi, Y)\alpha_\phi$ and $f(Y)$. In other words, the coefficients α_ϕ will be the solution of the following linear least-squares problem :

$$\min_{\alpha_\phi} \ \|M(\phi, Y)\alpha_\phi - f(Y)\|_2^2. \tag{2.13}$$

When the coefficient matrix has full column rank, the minimization problem (2.13) above has a unique solution given by solving the normal equations

$$M(\phi, Y)^\top M(\phi, Y)\alpha_\phi \ = \ M(\phi, Y)^\top f(Y).$$

To solve this linear system, singular value decomposition or QR factorization of the coefficient matrix can be used. Regression models are very recommended to use, especially when the objective function is noisy [50, 52].

2.1.3 An interpolation based trust-region approach

Different interpolation-based trust-region methods are available in the literature. The existing methods can be divided into two categories, the first one being the methods that work well for practical problems but are not supported by a convergence theory. The second category includes the methods for which global convergence was shown, but that are practically less competitive than the first category. The algorithm framework which will be described in this section requires the usage of fully linear models, meaning models with accuracy properties similar to those of first-order expansion Taylor model. A rigorous definition of a fully linear model can be found in [51, Definition 3.1] (see also [52, Definition 10.3]). Algorithm 2.1 a derivative-free interpolation based trust-region algorithm for which global convergence to first-order stationary points is proved [51, 52].

The algorithm as presented is simple, we check if the norm of the model gradient is too small. If it is, we start the criticality step with the purpose of verifying if the gradient of the objective function f is also small. At each iteration, many situations can occur: an iteration is **successful** whenever $\rho_k \geq \nu_1$; the trial point is then accepted and the trust-region radius is increased by a factor $\gamma_{inc} > 1$ or kept the same. When

Algorithm 2.1: A DFO trust-region algorithm.

Initialization: Let an initial point x_0 and the value $f(x_0)$ be given. Choose an initial trust-region radius $\Delta_0 > 0$. Select an initial model m_0. Set $k = 0$ and the parameters $\epsilon_g > 0$; $0 < \gamma < 1 < \gamma_{inc}$, $0 < \nu_0 \leq \nu_1 < 1$, $\mu > \beta > 0$.

1. Criticality step : Apply some procedure when $\|\nabla m_k(x_k)\| \leq \epsilon_g$ to find a new model m_k and a new trust region radius Δ_k such that $\Delta_k \leq \mu\|\nabla m_k(x_k)\|$ and m_k is fully linear on $B(x_k; \Delta_k)$, and such that, if Δ_k is reduced, one has $\beta\|\nabla m_k(x_k)\| \leq \Delta_k$.

2. Compute the step : Compute a step s_k such as

$$s_k \;=\; \text{argmin}_{s \in B(0,\Delta_k)}\, m_k(x_k + s). \tag{2.14}$$

2. Accept the trial point :

Compute $f(x_k + s_k)$ and

$$\rho_k \;=\; \frac{f(x_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)}.$$

If $\rho_k \geq \nu_1$ or if both $\rho_k \geq \nu_0$ and the model is fully linear on $B(x_k; \Delta_k)$, then $x_{k+1} = x_k + s_k$ and the model is updated to take into consideration the new iterate, resulting in a new model m_{k+1}; otherwise $m_{k+1} = m_k$ and $x_{k+1} = x_k$.

4. Improve the model :

If $\rho_k < \nu_1$ use a model-improvement algorithm to certify that the model m_k is fully linear on $B(x_k, \Delta_k)$. Let m_{k+1} the new possibly improved model.

5. Update the trust-region radius: Set

$$\Delta_{k+1} = \begin{cases} [\Delta_k, \min\{\gamma_{inc}\Delta_k, \Delta_{max}\}] & \text{if} \quad \rho_k \geq \nu_1, \\ \gamma\Delta_k & \text{if} \quad \rho_k < \nu_1 \;\text{ and } m_k \text{ is fully linear}, \\ \Delta_k & \text{if} \quad \rho_k < \nu_1 \;\text{ and } m_k \text{ is not} \\ & \text{certifiably fully linear.} \end{cases}$$

Increment k by one and return to Step 1.

$\nu_0 \leq \rho_k < \nu_1$ and the model is fully linear (see Algorithm 2.1), the trial point is again accepted but the trust-region is decreased; such iteration is called **acceptable**. The third situation occurs when $\rho_k < \nu_1$ and the model m_k is not certifiably fully linear (see [51, Definition 3.1]). In this case, the geometry should be improved; the trial point may be included in the sample set but it will not accepted as the new iterate; such iteration is called **model-improving**. The last situation occurs when $\rho_k < \nu_0$ and m_k is fully linear, in this case only the trust-region radius is reduced, the other parameters (including the current iterate) are kept the same; such iteration is declared **unsuccessful**. The model-improvement cycle in Step 4 can be launched for an infinite number of iterations.

However, when the models are assumed to be fully linear and uniformly bounded, one can ensure that only finite improvement steps will take place [52]. The criticality step is not invoked in detail (see [51, 52] for more details), but mainly in such a step one keeps reducing the trust-region radius Δ_k and computes a fully linear model in $B(x_k; \Delta_k)$ until $\Delta_k \leq \mu\|\nabla m_k(x_k)\|$ is obtained. At the exit of the criticality step one also has $\Delta_k \geq \beta\|\nabla m_k(x_k)\|$ (with $\mu > \beta$).

2.1.3.1 The trust-region subproblem

In Step 2 of Algorithm 2.1, one needs to approximate a minimizer s_k of the following optimization problem (called trust-region subproblem):

$$\min_{s \in B(0,\Delta_k)} m_k(x_k + s), \tag{2.15}$$

where m_k is the model for the objective function and $B(0, \Delta_k)$ is the trust-region. The computation of such step s_k is crucial for the convergence theory of the trust-region methods. In general, it is not necessary to find an exact minimizer of this optimization problem as far as the computed step ensures some form of sufficient decrease condition, meaning that the new step s_k has to fulfill

$$m_k(x_k + s_k) \quad \leq \quad m_k(x_k) - \psi_k,$$

where ψ_k is a positive value satisfying suitable conditions [52]. The key point is to make sure that the total decrease is at least a fraction of that obtained with the Cauchy step s_k^C [52, Chapter 10], for all iterations k:

$$m(x_k) - m_k(x_k + s_k) \quad \geq \quad \kappa_{fcd}[m(x_k) - m_k(x_k + s_k^C)], \tag{2.16}$$

where $\kappa_{fcd} \in (0, 1]$. The Cauchy step s_k^C can be computed by backtracking a line search along the steepest descent direction given by the gradient of the model. As a consequence, the Cauchy step is defined by

$$s_k^C = -t_k^C g_k, \tag{2.17}$$

where t_k^C is given by

$$t_k^C = \underset{t \geq 0 : x_k - t g_k \in B_k(x_k, \Delta_k)}{\operatorname{argmin}} m_k(x_k - t g_k).$$

The Cauchy step satisfies the condition:

$$m_k(x_k) - m_k(x_k + s_k^C) \quad \geq \quad \frac{1}{2}\|g_k\| \min\left\{\frac{\|g_k\|}{\|H_k\|}, \Delta_k\right\}. \tag{2.18}$$

2.1.3.2 Global convergence

To prove global convergence to first-order critical points (convergence to a stationary point regardless the starting point), it suffices to assume in addition to the assumption (2.16), that the gradient of the objective function f is Lipschitz continuous. We suppose also that the Hessian model is bounded (see [52] for a complete and detailed convergence analysis).

Under such assumptions it is provable that the trust-region radius in Algorithm 2.1 converges to zero [52, Lemma 10.9]:

Lemma 2.1. *Consider a sequence of iterations generated by Algorithm 2.1 without any stopping criterion. Then under the above assumptions one has*

$$\lim_{k \to +\infty} \Delta_k = 0. \tag{2.19}$$

When the sequence of iterates is bounded, one can also prove that all limit points of the sequence of iterates are first-order stationary points. The global convergence result is then derived as follows [52, Theorem 10.13]:

Theorem 2.2. *Consider a sequence of iterations generated by Algorithm 2.1 without any stopping criterion. Then under the above assumptions one has*

$$\lim_{k \to +\infty} \nabla f(x_k) \; = \; 0. \tag{2.20}$$

2.2 Direct-search methods

Direct-search methods correspond to DFO algorithms where sampling, at each iteration, is guided by a finite set of directions with some appropriate features. These methods do not use any derivative approximation or model building. In this section, by direct-search we mean the directional type; we refer the reader to [52, 102, 128] and references therein for more details on the other types of direct-search methods. To describe direct-search algorithms, we first present some related basic concepts.

2.2.1 Basic concepts

To guide the optimization process, the directions used in direct-search methods must have some appropriate features. One essential property consists on ensuring that at least one of the chosen directions is descent. A direction d is said to be descent at the point x, if there exists a positive value $\bar{\alpha}$ such that:

$$\forall \alpha \in (0, \bar{\alpha}] \quad , \quad f(x + \alpha d) < f(x). \tag{2.21}$$

When f is continuously differentiable at x and $\nabla f(x) \neq 0$, all the descent directions d fulfill $-\nabla f(x)^\top d > 0$. To ensure the existence of such directions, some notions related to positive spanning sets and positive bases are needed [52, 56].

2.2.1.1 Positive spanning sets and positive bases

The positive span of a set (PSS) of vectors $[v_1, \ldots, v_r]$ in $\mathbb{R}^n$ is defined as the convex cone which is positively generated by $[v_1, \ldots, v_r]$ (meaning the set $\{v \in \mathbb{R}^n : \ v = \sum_{i=1}^{r} \alpha_i v_i, \ \alpha_i \geq 0, \ i = 1, \ldots, r\}$) [52, 56].

Definition 2.3.

- A positive spanning set in $\mathbb{R}^n$ is a set of vectors whose positive span is $\mathbb{R}^n$.
- The set $[v_1, \ldots, v_r]$ is said to be positively dependent, if one of the vectors is in the convex cone positively spanned by the remaining vectors, i.e, if one of the vectors is a positive combination of the others; otherwise, the set is positively independent.
- A positive basis in $\mathbb{R}^n$ is a positively independent set whose positive span is $\mathbb{R}^n$.

Unlike $\mathbb{R}^n$ bases where one has exactly n vectors, the cardinality of a positive basis has at least $n+1$ and at most $2n$ vectors [15, 56]. Positive bases with $n+1$ and $2n$ vectors are referred to as the minimal and the maximal positive bases, respectively.

Example 2.1. *Let $B = [e_1, e_2, \ldots, e_n]$ be the canonical basis of $\mathbb{R}^n$, where e_i denotes the vector with a 1 in the i^{th} coordinate and 0's elsewhere, and let $e = \sum_{i=1}^{n} e_i$, then*

- *$D_\oplus = [B \ , \ -B]$ is a maximal positive basis of $\mathbb{R}^n$, where $-B = [-e_1, -e_2, \ldots, -e_n]$.*
- *$[B \ , \ \frac{-e}{\|e\|}]$ is a minimal positive basis.*

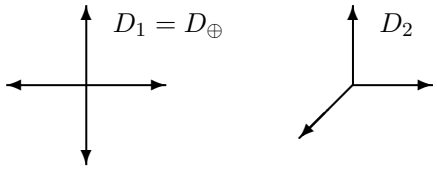


FIGURE 2.1: A graphical representation of the maximal positive basis D_1 (left) and the minimal positive basis D_2 (right) for $\mathbb{R}^2$.

In Figure 2.1, we depict two positive bases for $\mathbb{R}^2$ (maximal and minimal).

As stated in [52, Theorem 2.4], if $[v_1, \ldots, v_r]$ is a positive basis for $\mathbb{R}^n$ and $W \in \mathbb{R}^{n\times n}$ is a nonsingular matrix, then $[Wv_1, \ldots, Wv_r]$ is also a positive basis for $\mathbb{R}^n$. In other words, having a positive basis in $\mathbb{R}^n$, one can ensure the existence of infinitely many different ones. Attractive properties of positive bases (explaining their use in direct-search methods) are as follows:

Theorem 2.4. *Let $[v_1, \ldots, v_r]$ be a positive basis for $\mathbb{R}^n$ and $w \in \mathbb{R}^n$. then*

$$\left[\forall i \in \{1, \ldots, r\} \quad v_i^\top w \geq 0\right] \Rightarrow \left[w = 0\right]. \tag{2.22}$$

Proof. Since $[v_1, \ldots, v_r]$ spans $\mathbb{R}^n$ positively, the vector $-w$ can be written as

$$-w \;=\; \sum_{i=1}^{r} \lambda_i v_i,$$

where each $\lambda_i \geq 0$ for all $i = 1, \ldots, r$.

From (2.22) we have $v_i^\top w \geq 0$ for all $i \in \{1, \ldots, r\}$ and so

$$0 \;\leq\; \sum_{i=1}^{r} \lambda_i v_i^\top w \;=\; -w^\top w \;\leq\; 0.$$

The only possibility is then $w = 0$. □

Thus by choosing $w = -\nabla f(x)$ in Theorem 2.4, positive bases can be used to check either a point $x \in \mathbb{R}^n$ is a stationary point of the objective function or not.

Theorem 2.5. *Let f be a continuously differentiable function with $\nabla f(x) \neq 0$ for some $x \in \mathbb{R}^n$. Let $[v_1, \ldots, v_r]$ be a positive basis for $\mathbb{R}^n$, then there exists i in $\{1, \ldots, r\}$ such as*

$$-\nabla f(x)^\top v_i \;>\; 0.$$

Proof. Let $w = -\nabla f(x)$ where $x \in \mathbb{R}^n$. one knows that $w^\top w > 0$ for all non-zero w and since $[v_1, \ldots, v_r]$ spans $\mathbb{R}^n$ positively, one has

$$w = \sum_{i=1}^{r} \lambda_i v_i,$$

where each $\lambda_i \geq 0$ for all $i = 1, \ldots, r$. Hence,

$$w^\top w = \sum_{i=1}^{r} \lambda_i w^\top v_i > 0$$

from which we conclude that at least one of the scalars $w^\top v_1, \ldots, w^\top v_r$ has to be positive. □

In other words, Theorem 2.5 states that there must exist at least one descent direction in a positive basis. In Figure 2.2, we identify the descent direction for the two positive spanning sets D_1 and D_2 in $\mathbb{R}^2$.

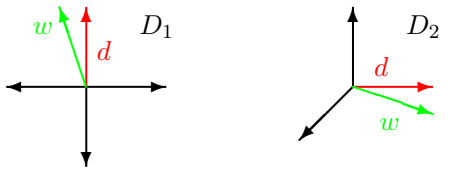


FIGURE 2.2: For a given positive spanning set and a vector $w = -\nabla f(x)$ (green), there must exist at least one descent direction d (red) (i.e. $w^\top d > 0$).

2.2.1.2 Gradient estimates

By assuming that the set of search directions is a PSS, one is sure that for each iteration a descent direction must exist in the PSS. However, in practice finding a good descent direction may not be possible, see for instance Figure 2.3 where two vectors of the PSS tend to be colinear opposite. A good descent direction can be defined as a direction

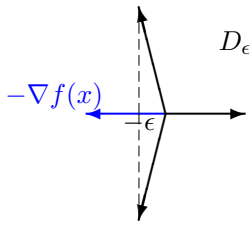


FIGURE 2.3: A positive spanning set with a very small cosine measure.

leading to a sufficient decrease of the objective function, which can be interpreted as:

the more acute the angle between the descent direction and the negative gradient of the objective function, the better the direction. A PSS gives descent directions at each iteration but may not be good enough (depending on the level of acuteness) to ensure convergence; in this case the PSS is said to be degenerate. Thus, the question that arises naturally is: how to measure and control any deterioration in the PSS property to avoid its degeneracy ? For that sake, we review the notion of the cosine measure for positive spanning sets [108].

Definition 2.6. The cosine measure of a positive spanning set (with nonzero vectors) or of a positive basis D is defined by

$$\mathrm{cm}(D) \;=\; \min_{0\neq v\in\mathbb{R}^n} \max_{d\in D} \frac{v^\top d}{\|v\|\|d\|}.$$

In $\mathbb{R}^2$, the cosine measure of a positive spanning set is the cosine of the half of the largest angle θ between two of its adjacent vectors (see Figure 2.4).

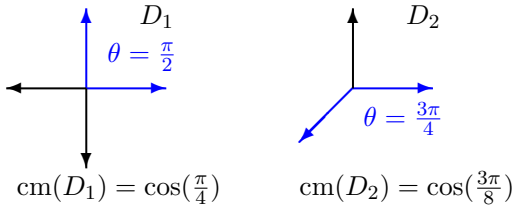


FIGURE 2.4: In $\mathbb{R}^2$, for a given positive spanning set the cosine measure is defined by $\cos(\theta)$ where θ (blue) is the largest angle between two adjacent vectors.

Remark 2.7. The cosine measure of a positive set is strictly positive.

In terms of descent, a key point of the cosine measure can be seen as follows: given a nonzero vector $w \in \mathbb{R}^n$, one has

$$\mathrm{cm}(D) \;\leq\; \max_{d\in D} \frac{w^\top d}{\|w\|\|d\|}.$$

Thus there must exist a $d \in D$ such that

$$\mathrm{cm}(D) \;\leq\; \frac{w^\top d}{\|w\|\|d\|}.$$

In particular if one chooses $w = -\nabla f(x)$, then

$$\mathrm{cm}(D)\|\nabla f(x)\|\|d\| \;\leq\; -\nabla f(x)^\top d. \tag{2.23}$$

A cosine measure close to zero indicates a deterioration of the PSS, meaning that the PSS becomes degenerate. To see how the cosine measure can predict such deterioration,

we emphasize the following example. Suppose that one has the following PSS :

$$D_\epsilon = \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -\epsilon \\ -1 \end{pmatrix}, \begin{pmatrix} -\epsilon \\ 1 \end{pmatrix} \right\}.$$

where $\epsilon > 0$. The cosine measure of this set is $\frac{\epsilon}{\sqrt{1+\epsilon^2}}$, then as ϵ tends to zero the cosine measure $\mathrm{cm}(D_\epsilon)$ goes to zero also. If $\nabla f(x) = (1 \quad 0)^\top$, as shown in Figure 2.3, then the quality of the descent directions in D_ϵ is poor (for small values of ϵ) and the lower bound of (2.23) is small compared to $\|\nabla f(x)\|$.

To avoid such situations, the cosine measure must be uniformly away from zero; that is,

$$\exists \xi > 0; \forall \epsilon \in (0, +\infty); \mathrm{cm}(D_\epsilon) \geq \xi \tag{2.24}$$

Such an assumption limits the deterioration of the positive spanning set and will be also important to the analysis of the global convergence of direct-search methods (described in the next section). We provide in the following example some values of the cosine measure for known positive bases.

Example 2.2.

- *If* $D = D_\oplus$, *then* $\mathrm{cm}(D) = \frac{1}{\sqrt{n}}$.
- *If* D *is a positive basis with* $n+1$ *elements uniformly distributed (the same angle between any two adjacent vectors), then* $\mathrm{cm}(D) = \frac{1}{n}$.

Based on the values of the cosine measure given in Example 2.2, one can explain why the performance of direct-search methods may deteriorate for large scale optimization problems, since as far as n grows the cosine measure goes to zero, and so the assumption (2.24) does not hold anymore.

2.2.2 Direct-search methods

Direct-search methods are derivative-free methods for which each iteration is based on the evaluation of the objective function at a finite set of points obtained from moving along a PSS [17, 52].

2.2.2.1 Coordinate-search method

Coordinate-search method is a direct-search method that uses the maximal positive basis $D_\oplus$ as PSS. An iteration of the algorithm can be described as follows. Let x_k be the

current iterate and α_k the associate step size. One evaluates the objective function f at the following points

$$P_k \;=\; \{x_k + \alpha_k d : d \in D_{\oplus}\}$$

to find a point that decreases the objective function value. This step of evaluating the objective function is called the **polling step** [36], the set P_k is known as the set of poll points and $D_{\oplus}$ is the set of poll directions.

Figure 2.5 shows the polling process for a coordinate-search method. At each iteration two situations are possible. The first is the **successful** iteration, meaning that a point in the polling set P_k is found to be better than the current iterate x_k. In this case, the new iterate $x_{k+1} = x_k + \alpha_k d_k \in P_k$ should achieve a simple decrease in the objective function (i.e. $f(x_{k+1}) < f(x_k)$). The step size α_{k+1} of a successful iteration is either left unchanged or increased by a factor $\gamma \geq 1$. For instance, in Figure 2.5, the first four iterations are all successful. The second possible situation occurs when no point, in the polling set P_k, ensures a simple decrease in the objective function. In this case, the step size α_k is reduced by a factor $\beta < 1$ and the current iterate is kept unchanged. Such iteration is declared **unsuccessful**, see for instance the fifth iteration in Figure 2.5. The evaluation process of the objective function, can be done following different strategies, opportunistically by moving towards the first evaluated point better than the current iterate (see Figure 2.5), or in a complete way, by evaluating all the poll points and choose the best point that improves the objective function.

Algorithm 2.2: Coordinate-search method.

Initialization: Let an initial point x_0 and choose an initial step size $\alpha_0 > 0$. Set $k = 0$ and the parameters $0 < \beta < 1 \leq \gamma$.

Until some stopping criterion is satisfied:

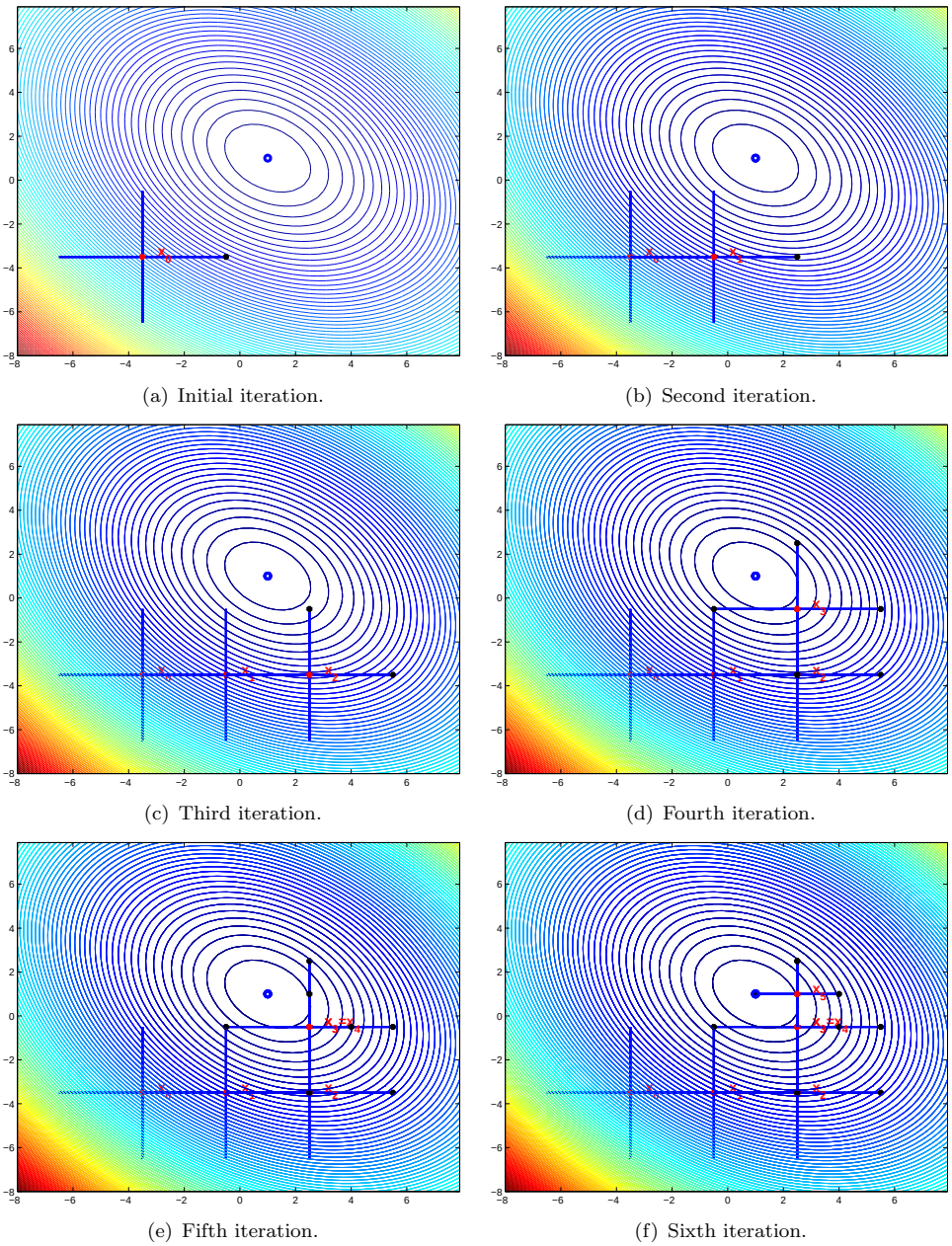
1. **Poll step:** Evaluate the objective function f at the polling set points P_k following the chosen evaluation process (opportunistic or complete).

   If a poll point $x_k + \alpha_k d_k$ is found such that $f(x_k + \alpha_k d_k) < f(x_k)$, then set $x_{k+1} = x_k + \alpha_k d_k$ and declare the poll (and the iteration) as successful.

   Otherwise, set $x_{k+1} = x_k$ and declare the poll (and the iteration) as unsuccessful.

2. **Update the step size parameter:** If the iteration is successful, then set $\alpha_{k+1} = \alpha_k$ (or $\alpha_{k+1} = \gamma\alpha_k$). Otherwise, set $\alpha_{k+1} = \beta\alpha_k$. Increment k by one and return to Step 1.

The performance of Algorithm 2.2 can be significantly enhanced through an optional step called a **search step** [36]. The latter one consists of using the previously evaluated points to find a new point y such that $f(y) < f(x_k)$. If the search step is successful, the



(a) Initial iteration. (b) Second iteration.

(c) Third iteration. (d) Fourth iteration.

(e) Fifth iteration. (f) Sixth iteration.

FIGURE 2.5: Six iterations of the coordinate-search method with opportunistic polling (following the order East/West/North/South). The initial point is $x_0 = [-3.5, -3.5]$, the starting step size is $\alpha_0 = 3$. For successful iterations, the step size is kept unchanged, otherwise it is reduced by a factor $\beta = 1/2$. The ellipses show the level sets of the objective function $f(x) = (x_1 + x_2 - 2)^2 + (x_1 - x_2)^2$. The optimum is located at the point $[1, 1]$.

iteration is declared successful, the poll step is skipped and $x_{k+1} = y$. The use of the search step is for practical reasons and has no interferences in the global convergence property. The next section will describe a general framework for direct-search methods including coordinate-search method.

2.2.2.2 Direct-search framework

In this section, we outline a general algorithmic description of direct-search methods, such description includes the previous framework [18, 52] (based on *integer lattice* and only simple decrease on the objective function value to compute the new iterate) as well as direct-search methods based on randomly generated directions but with sufficient decrease condition to identify the new iterate [166]. To define the type of sufficient decrease conditions we are using, we introduce the following notion of a forcing function [108]:

Definition 2.8. We call a non-decreasing continuous function $\rho : \mathbb{R}_+^* \to \mathbb{R}_+^*$ a forcing function if it satisfies

$$\lim_{t \to 0^+} \frac{\rho(t)}{t} \;=\; 0$$

One example of such forcing function is $\rho(t) = t^2$.

To describe the algorithm in the most general way, we will use $\bar{\rho}(.)$. The latter one will be equal to the forcing function $\rho(.)$ when the directions are randomly generated, or equal to the constant zero function when the directions rely on integer lattices (i.e, MADS [17]). Algorithm 2.3 gives a complete description of a typical direct-search algorithm. Its framework can be formulated in the same way as coordinate-search, where the basic idea of the algorithm relies on a polling step, in which we evaluate a set of points in order to improve sufficiently the current iterate. By sufficiently, we mean that the new point will be accepted only if a sufficient decrease condition is fulfilled. In other words, a new point $x_{k+1} \neq x_k$ is accepted only if

$$f(x_{k+1}) \;<\; f(x_k) - \bar{\rho}(\alpha_k \|d_k\|). \tag{2.25}$$

The new iterate x_{k+1} is found by exploring a set of points defined by a positive spanning directions set D_k and a step size parameter α_k:

$$P_k \;=\; \{x_k + \alpha_k d : d \in D_k\}, \tag{2.26}$$

The poll step and the iteration are declared successful, if a new point satisfying the condition (2.25) is found. In that case, the step size parameter is kept unchanged or possibly increased. When the poll step fails to find a new point x_{k+1}, the iteration is regarded as unsuccessful, the current iterate is kept the same and the step size parameter is reduced. Again, the search step [36] is optional and has no impact on the convergence properties of the algorithm. It takes benefit from the previously evaluated points to speed up convergence and make the algorithm more efficient. A new point y will be accepted only if it decreases sufficiently the objective function (i.e. $f(y) \;<\; f(x_k) - \bar{\rho}(\alpha_k \|d_k\|)$),

in such case the iteration is declared successful, $x_{k+1} = y$ and the polling step is skipped.

Algorithm 2.3: A direct-search method.

Initialization: Let an initial point x_0 and choose an initial step size $\alpha_0 > 0$. Set $k = 0$ and the parameters $0 < \beta_1 \leq \beta_2 < 1 \leq \gamma$.

Until some stopping criterion is satisfied:

1. **Search step:** Try to compute a point with $f(y) < f(x_k) - \bar{\rho}(\alpha_k \|d_k\|)$ by evaluating the objective function f at a finite number of points. If such point is found, then set $x_{k+1} = y$, declare the iteration and the search step successful, and skip the poll step.

2. **Poll step:** Choose a set D_k of directions in $\mathbb{R}^n$. Evaluate the objective function f at the polling set points P_k following the chosen evaluation process (opportunistic or complete).

   If a poll point $x_k + \alpha_k d_k$ is found such that $f(x_k + \alpha_k d_k) < f(x_k) - \bar{\rho}(\alpha_k \|d_k\|)$, then set $x_{k+1} = x_k + \alpha_k d_k$ and declare the poll (and the iteration) as successful.

   Otherwise, set $x_{k+1} = x_k$ and declare the poll (and the iteration) as unsuccessful.

3. **Update the step size parameter:** If the iteration is successful, then set $\alpha_{k+1} \in [\alpha_k, , \gamma\alpha_k]$. Otherwise, set $\alpha_{k+1} \in [\beta_1\alpha_k, , \beta_2\alpha_k]$. Increment k by one and return to Step 1.

2.2.3 Global convergence

The global convergence of direct-search methods, outlined by Algorithm 2.3, relies on proving that the behavior of the step size parameter α_k will approach zero as an indicator of some form of stationarity. Such result can be established using two different strategies: the first one requires the iterates to lie on integer lattices (known as pattern search) [52, 108, 163]. The second strategy consists in imposing a sufficient decrease condition on the objective function values to accept or not the new iterate [52, 166]. In this thesis, only the global convergence theory related to the second strategy is outlined. The reader is referred to the references [17, 18, 52, 108, 163] for the convergence theory when one is requiring the iterates to lie on integer lattices.

Direct-search methods are traditionally analyzed under the assumption that all the iterates lie in a bounded set and that the objective function is bounded below.

Assumption 2.2.1. The level set $L(x_0) = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ is bounded. The objective function is bounded below.

Moreover, the following assumption is also needed:

Assumption 2.2.2. The distance between x_k and the point $x_k + \alpha_k d_k$ tends to zero if and only if α_k does:

$$\lim_{k \in K} \alpha_k \|d_k\| = 0 \Leftrightarrow \lim_{k \in K} \alpha_k = 0,$$

for any infinite subsequence K.

Such assumption can be fulfilled, if one chooses to work with random directions generated in the unit sphere (i.e. $\|d_k\| = 1$).

By imposing the condition $f(x_k + \alpha_k d_k) < f(x_k) - \bar{\rho}(\alpha_k \|d_k\|)$, the former assumptions lead the step size to converge to zero.

The sufficient decrease condition suffices to ensure that the step size parameter, as defined by Algorithm 2.3, converges to zero [52, 108] as follows:

Theorem 2.9. *Let Assumptions 2.2.1 and 2.2.2 hold. Consider Algorithm 2.3 when* $\bar{\rho}(.) = \rho(.)$. *Then there exists a subsequence* K *of unsuccessful poll steps such that*

$$\lim_{k \in K} \alpha_k = 0.$$

Since $L(x_0)$ *is bounded (Assumption 2.2.1), there exist a point* x_* *and a subsequence* K *of unsuccessful iterations such that* $\lim_{k \in K} \alpha_k = 0$ *and* $\lim_{kinK} x_k = x_*$.

2.2.3.1 Global convergence for smooth functions

By imposing a sufficient decrease condition, one is able to derive stationarity results in the continuously differentiable case. But, before, we need to assume first that the search directions in the poll step has to positively span the whole space and that the cosine measures of such set is bounded away from zero.

Assumption 2.2.3. For all k, the set D_k used for the polling has to be a positive spanning set (PSS) and must satisfy $\text{cm}(D_k) \geq \xi$ with $\xi > 0$.

As observed originally in [108], global convergence can be derived as follows:

Theorem 2.10. *Let Assumptions 2.2.1 and 2.2.2 hold. Consider Algorithm 2.3 under Assumption 2.2.3. Assume also that* f *is continuously differentiable with Lipschitz continuous gradient on an open set containing* $L(x_0)$. *Then, there exists a subsequence* K *of unsuccessful poll steps such that* $\lim_{k \in K} \alpha_k = 0$. *and*

$$\lim_{k \in K} \nabla f(x_k) = 0.$$

Since $L(x_0)$ *is bounded (Assumption 2.2.1), there exists a point* x_* *such that* $\nabla f(x_*) = 0$.

2.2.3.2 Global convergence for non-smooth functions

The only major difference compared to the smooth case is that the search directions in the poll step do not need to positively span the whole space. We introduce first some basic notions for non-smooth optimization to outline the global convergence properties of direct-search methods based on the sufficient decrease strategy [166].

The first concept is related to the stationarity results performed at limit points of specific subsequences known as refining subsequences [17]. More concepts will be outlined in relation with the non-smooth calculus [43] used to analyze Algorithm 2.3. A refining subsequence can be formalized as a sequence of unsuccessful iterates driving the step size to zero [17]. Theorem 2.9 states that the convergence properties of direct-search methods are derived only for refining subsequences.

Definition 2.11. A subsequence $\{x_k\}_{k\in K}$ of iterates corresponding to unsuccessful poll steps is said to be a refining subsequence if $\lim_{k\in K} \alpha_k = 0$.

The type of directions along which a directional derivative will be proved nonnegative are the so-called refining directions [17].

Definition 2.12. Let x_* be a limit point of a convergent refining subsequence K. If the $\lim_{k\in L} d_k/\|d_k\|$ exists, where $L \subset K$ and $d_k \in D_k$ then this limit is said to be a refining direction for x_*.

Assuming that the objective function f is Lipschitz continuous near x_*. The Clarke generalized directional derivative [43] at x_* along the direction d is defined by

$$f^\circ(x_*; d) \;=\; \limsup_{x\to x_*, t\downarrow 0} \frac{f(x+td) - f(x)}{t}.$$

The following results are showing that the Clarke generalized directional derivative is Lipschitz continuous with respect to the second argument[43]:

Proposition 2.13. *Let f be a Lipschitz continuous near x_* with constant L_f. Then the function $d \to f^\circ(x_*; d)$ is Lipschitz continuous in $\mathbb{R}^n$ with constant L_f.*

The Clarke subdifferential is defined by

$$\partial f(x_*) = \{s \in \mathbb{R}^n : f^\circ(x_*; d) \geq \langle d, s\rangle, \forall d \in \mathbb{R}^n\}, \tag{2.27}$$

where $\langle ., .\rangle$ denotes the dot product of two vectors. When the function f is smooth, a point x_* is said to be stationary point if $\nabla f(x_*) = 0$. In the non-smooth case, the stationarity is defined as follows:

Definition 2.14. Let f be a Lipschitz continuous near x_*. A point x_* is said to be Clarke stationary if $f^\circ(x_*; d) \geq 0$, $\forall d \in \mathbb{R}^n$, or, in other words, $0 \in \partial f(x_*)$.

Definition 2.15. A function f is strictly differentiable at x_* if f is Lipschitz continuous near x_* and for some $\xi = \nabla f(x_*), \forall d \in \mathbb{R}^n, f^\circ(x_*; d) = \langle \xi, d \rangle$.

If f is strictly differentiable function and x_* is a Clarke stationary point, then $\nabla f(x_*) = 0$. As consequence, the convergence results in the smooth case can be seen as a particularization of the ones obtained using the Clarke calculus in the non-smooth case.

Under appropriate assumptions, the Clarke generalized derivative can be proved to be nonnegative along any refining direction for x_*. When the sequence of refining directions for x_* is dense in the unit sphere, one can conclude that x_* is a Clarke stationary point [52, 166].

Theorem 2.16. *Let Assumptions 2.2.1 and 2.2.2 hold. Consider a refining subsequence $\{x_k\}_{k \in K}$, generated by Algorithm 2.3 and converging to x_*. Assume that f is Lipschitz continuous near x_*. Then,*

$$f^\circ(x_*; d) \geq 0$$

for all refining directions d for x_*.

If the set of refining directions for x_* is dense in the unit sphere, then x_* is a Clarke stationary point of the objective function f.

2.3 Conclusion

In this chapter, we presented the main ideas, techniques, and algorithms used in deterministic DFO methods. This overview is given in an attempt to prepare the reader to what comes next. Chapter 3 will present stochastic DFO and more particularly evolution strategies (ES's), on which we will try to incorporate some of the techniques presented in this chapter to ensure its global convergence and enhance the original performance (see Chapter 4).

The model-based techniques presented in Section 2.1.2 will be used later to hybridize them with evolution strategies. In fact, by incorporating a search step at the beginning of each iteration, one expects to improve the algorithm efficiency and its convergence speed (as in the search-poll framework of direct search, see Section 2.2.2). In such a step, one can, for instance, build a quadratic model using all or some of the points where the objective function has been previously evaluated and then minimize such a model in a certain region.

Chapter 3

Stochastic Derivative-Free Optimization & Evolution Strategies

The early development of stochastic derivative free optimization methods was motivated mainly by the need for methods that mitigate the defect of the deterministic ones for hard optimization problems [122, 158]. The key idea of the introduction of randomness can be implemented through two different approaches. The first one is known as localized random search methods, where we construct an oriented path, starting from an arbitrary point, and then apply some stochastic decisions to obtain the new point. The second approach, known as volume oriented methods, contrary to the first one is based on the fact that the whole search space must be sampled, consequently, this approach is been seen as performing global search. In general, the main classes of stochastic optimization methods are as follows:

1. **Evolutionary Algorithms (EAs)** [26], where the optimization process is inspired by biological evolution. Its basic idea is to evolve a population of candidate solutions (individuals) using operators inspired by natural selection and genetic variation. The selection process focuses the search to "better" zones (which improves the objective function value) by encouraging individuals with a better function value to be a member of the next generation. Genetic variation is the second operation that creates new individuals in the search space. During the second process, one generally uses random changes of some particular points (mutation) and mixing of information of individuals (recombination). The different mechanisms used for natural selection and genetic variation give birth to many classes of EAs such as :

- Genetic Algorithms (GAs) [91, 92] were initially designed by Holland to cope with binary encoded individuals. In the continuous case, the variables are generally mapped to binary strings which sometimes leads to weak performance [89]. Some successful practical application of GAs are reported in [42].
- Evolution Strategies (ES's) were originally developed in [142, 150] and have been widely investigated and tested (see, e.g. [30, 32] and the references therein). In a large class of ES a certain number λ of points (called offspring) are randomly generated in each iteration, among which $\mu \leq \lambda$ of them (called parents) are selected. ES's have been growing rapidly in popularity and start to be used for solving challenging optimization problems [21, 79]. One well known instance of ES's is Covariance Matrix Adaptation ES (CMA-ES) [85, 86]. More details about ES's and CMA-ES will be provided later in this Chapter.
- Evolutionary Programming (EP) [64, 65] is similar to ES's and relies on mutation as a variation operator. The selection operator is a mixture of tournament selection and truncation selection. By tournament selection, we mean that the individuals are randomly chosen from the population. The truncation selection means that only a fraction of the best individuals is chosen. A relevant instance of EP is meta-EP [65] where one use a self-adaptation process to guide the population, similarly to the ES's.
- Differential Evolution (DE) [159], Learning Classifier System (LCS) [39] and Neuro-Evolution (NE) [76] algorithms are also considered as instances of EA's.

2. **Particle swarm optimization (PSO)** [103] is inspired by the movement of swarms of birds or insects searching for food or protection. The movement of each particle depends on both its local best known position and also the best known global position (found by other particles). Such process is expected to move the swarm toward the best solutions. An instance of PSO is PSWARM [164, 165], where one combines pattern search and particle swarm. Basically, it applies a directional direct search in the poll step (coordinate search in the pure simple bounds case) and particle swarm in the search step (see Section 2.2.2 for the definition of the search and poll steps).

3. **Simulated Annealing (SA)** [107] is inspired by the physical behavior of material during the annealing process. The latter is performed by controlling the material cooling to obtain regular crystals and push the system to end up with a minimum of energy. By analogy, this physical process is translated to the following algorithm: given a candidate solution, a neighbor random solution can be accepted (to replace the candidate solution) if the neighbor solution is better that the candidate one in

terms of the objective function or with a probability that depends on the change of the corresponding objective function values and a control parameters, called the temperature. When none of the above conditions are fulfilled, the current solution is unchanged and the temperature parameter is gradually decreased to zero. A relevant instance of AS is Adaptive Simulated Annealing (ASA) [94] where one starts from a traditional simulated annealing in which a different probability density function is used for each variable with separate temperature parameters. Such process allows ASA to possibly escape local minima.

This chapter gives an overview of the ES algorithms, their origin and history, their basic ideas and philosophy. It is organized as follows. The first section is intended to provide deeper insight into the basic ideas and principles as well as the ingredients for designing ES algorithms, such as mutation, recombination, and selection operators. The second section is devoted to emphasize theoretical aspects of ES research. In particular, the existing global convergence properties of ES algorithms. The chapter closes with a detailed description of the CMA-ES method which is regarded as the state of the art in stochastic derivative free optimization [145].

3.1 Evolution strategies

ES algorithms are firstly developed by Rechenberg and Schwefel [142, 150] in the early 1970s. From the beginning ES's were designed to solve real and integer optimization problems. The selection and the mutation mechanisms as well as the population concept are all described by the conventional notation $(\mu/\rho \stackrel{+}{,} \lambda)$–ES [30, 32], such notation is introduced within a general ES framework in the following section.

3.1.1 Notation and algorithm

Evolution strategies try to optimize an objective function f with respect to an n-dimensional set of decision variables $y \in \mathfrak{Y}$, known in the ES's community as the object variables. The search space $\mathfrak{Y}$ can be the n-dimensional real space $\mathbb{R}^n$ [32] or the integer space $\mathbb{Z}^n$ [25].

At the k-th generation, ES's work with a population $\mathfrak{B}_k$ of individuals $\mathfrak{a}_k^l$. An individual $\mathfrak{a}_k^l$ is represented by a decision variable y_k^l (its position), its objective function value $f_k^l = f(y_k^l)$ (known as the fitness), and possibly a set of endogenous parameters s_k^l. The parameters s_k^l control the capacity of the strategy for adaptive evolution as one of the

particularities of the ES's (i.e. evolvability).

$$\mathfrak{a}_k^l \overset{def.}{=} (y_k^l, s_k^l, f_k^l). \tag{3.1}$$

A new population of λ individuals (called offspring), noted $\tilde{\mathfrak{a}}_k^l$, is generated from a set of μ parent individuals $\mathfrak{a}_k^l$. The offspring population contains λ individuals, denoted $\mathfrak{B}_k^o$, while the parent population, denoted $\mathfrak{B}_k^p$, contains μ individuals. To create a new offspring population, one uses ρ parents, where ρ is the mixing number. When $\rho = 1$ (known as cloning), no recombination is used, this case is usually denoted by (μ, λ) or $(\mu + \lambda)$ depending on the regarded selection strategy. The algorithmic description of $(\mu/\rho \overset{+}{,} \lambda)$–ES is outlined in Algorithm 3.1. The symbol '$\overset{+}{,}$' outlines the type of the selection used to create the new parent population. The different possible selection schemes are emphasized later in Section 3.1.3.

Algorithm 3.1: A general framework for $(\mu/\rho \overset{+}{,} \lambda)$–ES.

Initialization: Choose positive integers λ, μ and ρ such that $\lambda \geq \mu \geq \rho$. Initialize μ individuals $\mathfrak{a}_0^l = (y_0^l, s_0^l, f_0^l)$, $l = 1, \ldots, \mu$. Let $\mathfrak{B}_0^p := (\mathfrak{a}_0^1, \ldots, \mathfrak{a}_0^\mu)$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. Offspring Generation:

$$\begin{aligned}
\mathfrak{m}_k^l &:= \text{marriage}(\mathfrak{B}_k^p, \rho), \\
s_k^l &:= \text{s_recombination}(\mathfrak{m}_k^l), \\
y_k^l &:= \text{y_recombination}(\mathfrak{m}_k^l), \\
\tilde{s}_k^l &:= \text{s_mutation}(s_k^l), \\
\tilde{y}_k^l &:= \text{y_mutation}(y_k^l), \\
\tilde{f}_k^l &:= f(\tilde{y}_k^l),
\end{aligned}$$

for all $l = 1, \ldots, \lambda$. Let the new offspring population be

$$\mathfrak{B}_k^o := \{(\tilde{y}_k^l, \tilde{s}_k^l, \tilde{f}_k^l), \quad l = 1, \ldots, \mu\}.$$

2. Parent Selection:

If (the comma-selection type (μ, λ)) then

$$\mathfrak{B}_{k+1}^p := \text{selection}(\mathfrak{B}_k^o, \mu)$$

If (the plus-selection type $(\mu + \lambda)$) then

$$\mathfrak{B}_{k+1}^p := \text{selection}(\mathfrak{B}_k^o, \mathfrak{B}_k^p, \mu)$$

Increment k and return to Step 1.

Algorithm 3.1 can be described as follows: given a generation k, the parent population $\mathfrak{B}_k^p$ produces a new offspring population $\mathfrak{B}_k^o$. This production process begins with the marriage step using ρ individuals , denoted $\mathfrak{m}_k^l$, which are randomly chosen from the parent population of size μ. The choice of individuals for marriage is completely randomized and independent of the objective function f. After the marriage, the recombination process of individuals is launched (see Section 3.1.2). The offspring generation is completed with the mutation operator (see Section 3.1.4). The parent selection is then performed using the chosen selection mechanism (see Section 3.1.3).

3.1.2 Recombination mechanism

ES's recombination is inspired by natural sexual reproduction in order to increase the genetic diversity of the offspring. For $(\mu/\rho \overset{+}{,} \lambda)$–ES, the recombination operator uses information only from ρ individuals (selected using the marriage operator) to produce one offspring. Two recombination operators are possible depending on wether the search space is continuous or discrete, known as intermediate recombination and discrete recombination, respectively.

Intermediate recombination deals with all ρ married parents by computing a weighted mean of all of them. Let $(\mathfrak{a}_m)_{1\leq m\leq \rho}$ be the chosen ρ parent individuals. The new recombinant offspring individual $\mathfrak{a}$ is the computed as follows :

$$\mathfrak{a} \;=\; \frac{1}{\rho}\sum_{m=1}^{\rho} \omega_m \mathfrak{a}_m. \tag{3.2}$$

The weights used to compute the means belong to a simplex set $\{(\omega^1,\ldots,\omega^\rho) \in \mathbb{R}^\rho : \sum_{i=1}^{\rho} w^i = 1, w^i \geq 0, i = 1,\ldots,\rho\}$, and their values reflect the contribution of each of the parents in the weighted mean. The way the weights are chosen has an important impact on the efficiency of the algorithm [12]. The intermediate recombination procedure is well defined for real-valued search space $\mathbb{R}^n$, but in the discrete search space case, one may need to round y given in (3.2) to map the discrete domain.

Discrete recombination combines randomly parameters value from ρ married parents, the i^{th} component of the recombinant object y and s are set to the i^{th} component randomly (uniformly) selected from the parent individuals. This means that for $i = 1,\ldots,n$:

$$(y)_i \;= (y_{m_i})_i \qquad \text{and} \qquad (s)_i \;= (s_{m_i})_i, \tag{3.3}$$

where m_i is randomly chosen in $\{1,\ldots,\rho\}$.

3.1.3 Selection mechanism

The main purpose of the selection operator is to guide the generations towards better regions in terms of the objective function value. Thanks to the selection mechanism an ES proceeds following a natural evolution. The selection process is inspired from natural selection where some beings (animals and plants) have to be strong enough to get a chance to survive. The selection operator tries to ensure such natural paradigm for all the new parent population. In Algorithm 3.1, the new parent population for the next generation is produced by ensuring that only the μ best individuals from the population, at the k-th generation, will survive. This selection mechanism is known as a truncation selection. The new parent population is then as follows:

$$\mathfrak{B}_{k+1}^{p} := (\mathfrak{a}_{k}^{1:\gamma}, \ldots, \mathfrak{a}_{k}^{\mu:\gamma}), \tag{3.4}$$

the notation $\mathfrak{a}_{k}^{m:\gamma}$ means that one takes the m^{th} best individual out of γ individuals [11, 32].

As mentioned in Section 3.1.1, two different selection operators are possible, depending on whether or not the parent population, at the generation k, is included: the comma-selection, denoted by (μ, λ), and the plus-selection, denoted by $(\mu + \lambda)$, respectively. For comma-selection, the new parent population $\mathfrak{B}_{k+1}^{p}$ is chosen only from the offspring individuals $\mathfrak{B}_{k}^{o}$. In this case, the selection is performed based on $\gamma = \lambda$ individuals. The plus-selection takes into account both the the new offspring population $\mathfrak{B}_{k}^{o}$ and the old parents population $\mathfrak{B}_{k}^{p}$. In contrast to the first selection type, the plus-selection is performed using $\gamma = \mu + \lambda$ individuals. The comma-selection variant of the algorithm can be good for dynamic problem instances given its capability for continued exploration of the search space, whereas the plus-selection variation can be good for refinement and convergence. In fact, the plus-section ensures that only the best individuals survive so far, thus such selection can be seen as *elitist*. Elitism can be a sufficient condition to ensure the global convergence of the ES's.

3.1.4 Mutation mechanism

3.1.4.1 The concept

Beside the selection operator, the mutations are another important process for an ES. The mutation process is at the origin of the genetic variations. If the selection mechanisms try to exploit the objective function information to guide the search into to promising regions, the mutations try to use only the search space information from the

parent population (no information from objective function is exploited). Consequently, no function based preference from the selection process is taken into consideration. The mutations depend on the problem structure, therefore its difficult to establish a general methodology. Meanwhile, Beyer [30] suggests some rules that may help during the mutation design such as reachability, scalability, unbiasedness, and symmetry.

Reachability This rule ensures that any given parent individual state $\mathfrak{a}^p$, can be transformed into any other (finite) individual state $\tilde{\mathfrak{a}}^p$ in a finite time. An ES needs to fulfill the reachability requirement particularly for proving its global convergence.

Scalability The scalability for the mutations operator states that the search length (strength mutation) should be tunable in order to adapt the evolution to the properties of both the objective function and the search space, known as fitness landscape. The secret behind the scalability is *evolvability* of the ES which favor improvement steps by using a smooth evolutionary random path to adapt the fitness landscape towards the optimum solution [8]. The scalability is defined as the capacity of a system for adaptive evolution. Again, by *evolvability* we mean the ability of the ES to generate adapted population, and thereby evolve through natural selection.

Unbiasedness The main condition is that the mutations should introduce no bias. This assumption is shown to be equivalent to have a mutations operator following the maximum entropy principle, meaning that the mutations distribution which best represents the current state of knowledge is the one with the largest entropy [99]. In the real-valued search space $\mathbb{R}^n$, the maximum entropy principle if the normal distribution as mutation operator is chosen.

Symmetry This rule is strongly connected to the previous one, but not equivalent. It means that the mean of the changes introduced by the mutation distribution should be zero.

3.1.4.2 Example in real-valued search spaces

To explain more precisely the definition of the mutation operator, we consider the following example. The first requirement to fulfill is that the mutation distribution should follow the maximum entropy principle. As mentioned earlier, in the real-value search space, i.e, $\mathfrak{Y} = \mathbb{R}^n$, such a requirement is shown to be equivalent to work with normal

distributions [99].

$$\tilde{y} \;=\; y + \sigma d \tag{3.5}$$

with d is a random vector generated following multivariate normal distribution $\mathcal{N}(0, I_n)$ of mean zero and identity matrix as covariance matrix. In this case, $\tilde{y}$ obeys the density function

$$\mathbb{P}(\tilde{y}) = \frac{1}{\sqrt{2\pi}\sigma^n} \exp\left(- \frac{(\tilde{y} - y)^\top (\tilde{y} - y)}{2\sigma^2} \right). \tag{3.6}$$

As the expected change is zero, such a distribution is symmetric and introduces no bias

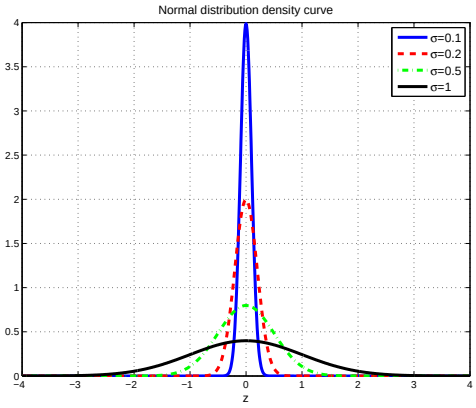


FIGURE 3.1: A scalar density function for a normal distribution.

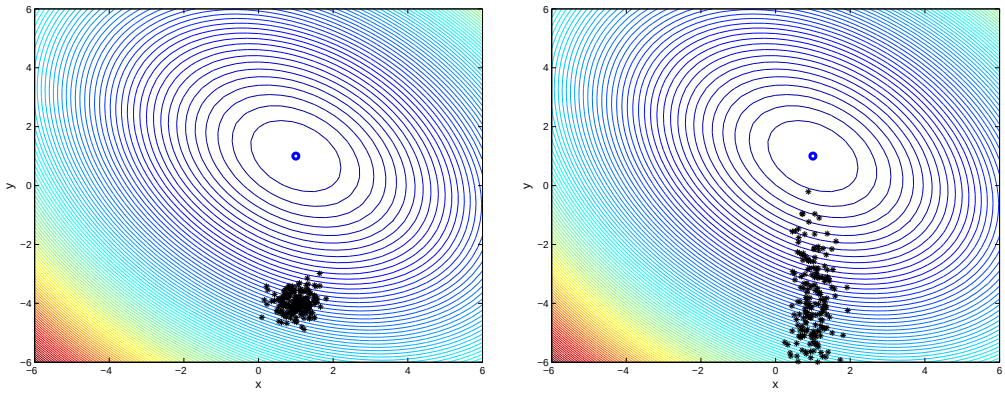
(see Figure (3.1)). Moreover, small perturbations around the point y are more likely to take place when the mutation strength σ is small. Thus, the scalability requirement can be fulfilled using a normal distribution. Based on the given mutations distribution, we need only one scalar parameter σ as endogenous parameter to control and adapt the evolution, such situation is known as isotropic mutations. Figure (3.2) depicts a 2-D situation where the adaptation process using a non-isotropic mutations can speed up the optimization process. It shows a simple case where axis-parallel mutations lead to a better exploration of the search space.

The mutation distribution can be improved if one has a proper evolution parameter σ_i for each component y_i of y using non-isotropic Gaussian mutations. The set of endogenous strategy parameters associated will be in this case an n-dimensional vector of standard deviation parameters $(\sigma_1, \ldots, \sigma_2)$. In this situation, the mutation distribution will be of the form

$$\tilde{y} \;=\; y + Sd, \tag{3.7}$$

where d is drawn from a normal distribution $\mathcal{N}(0, I_n)$ and S is a diagonal matrix.

The most general situation occurs when the mutation distribution can be also arbitrarily rotated in the search space. Figure (3.3) outlines a situation where the rotation, see



(a) Isotropic mutations.

(b) Non-isotropic mutations parallel to the y-axis.

FIGURE 3.2: A 2-D situation where non-isotropic mutations, parallel to the y-axis, enhance the performance. The ellipses show the level sets of the objective function $f(x) = (x_1 + x_2 - 2)^2 + (x_1 - x_2)^2$.

Figure (3.3(d)), can lead to better performance compared to both the isotropic mutations (Figure 3.3(a)) and non-isotropic (Figures (3.3(b)) and (3.3(c))). The rotation process actually reflects the distribution correlation between the z components, contrary to the assumption we made before where we assume that the components of the vector y are independent.

Let R be a rotation matrix, the new mutation distribution is of the form

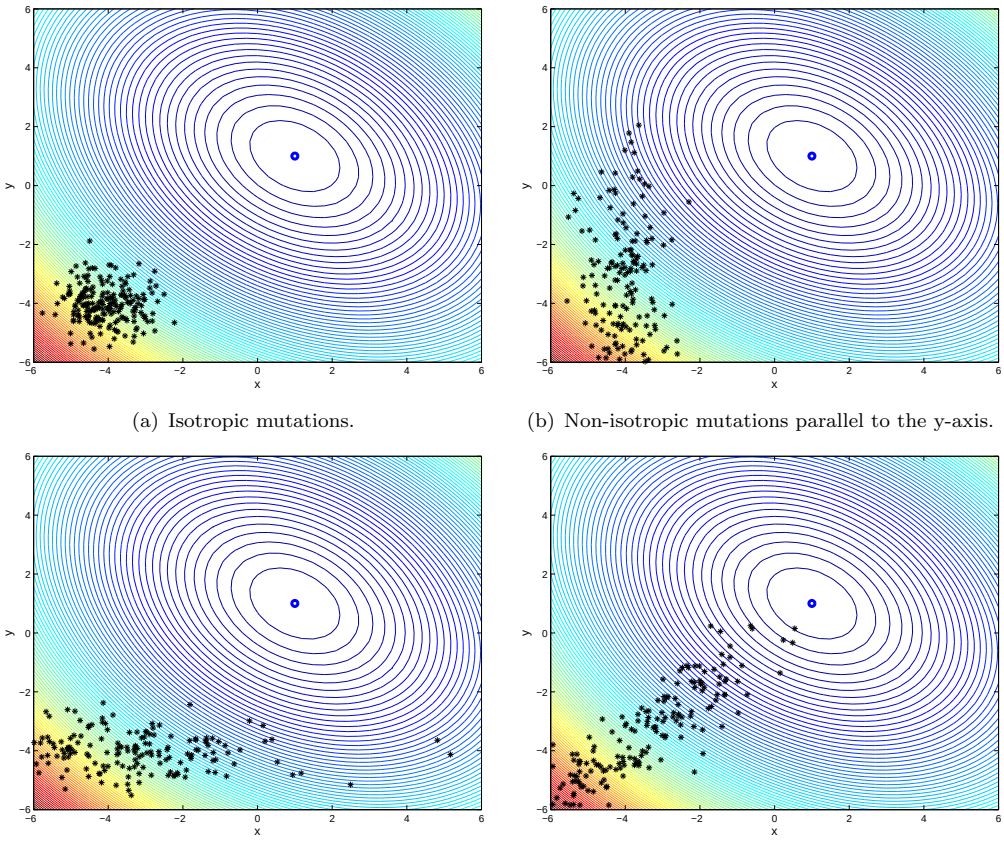
$$\tilde{y} \;=\; y + RSd, \tag{3.8}$$

Such equation is equivalent to assume that the mutated vector $\tilde{y}$ is drawn from a normal distribution of mean y and covariance matrix $C = RSS^\top R^\top$. Thus, the new density function of $\tilde{y}$ is as follows:

$$\mathbb{P}(\tilde{y}) = \frac{1}{\sqrt{2\pi}^n} \frac{1}{\sqrt{\det(C)}} \exp\left(-\frac{1}{2}(\tilde{y} - y)^\top C^{-1} (\tilde{y} - y) \right), \tag{3.9}$$

where $\det(C)$ corresponds to the determinant of matrix C.

Matrix C is symmetric, therefore only $n(n+1)/2$ endogenous strategy parameters are needed to define properly the mutation operator. Such an adaptation process for the mutation operator explains the success of the algorithm CMA-ES [86].



(a) Isotropic mutations.

(b) Non-isotropic mutations parallel to the y-axis.

(c) Non-isotropic mutations parallel to the x-axis.

(d) Rotated non-isotropic mutations.

FIGURE 3.3: A 2-D situation where it is more efficient to have correlated Gaussian mutations. The ellipses show the level sets of the objective function $f(x) = (x_1 + x_2 - 2)^2 + (x_1 - x_2)^2$.

3.2 A class of evolution strategies

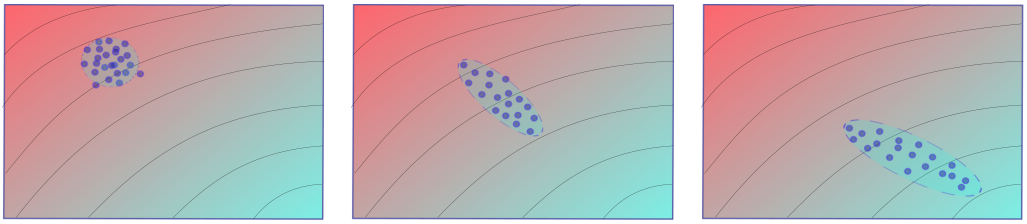
This section focuses only on a subclass of $(\mu/\rho \dagger \lambda)$–ES denoted by $(\mu/\mu_W, \lambda)$–ES in preparation to what comes next. In fact, all the contributions of this thesis are related to $(\mu/\mu_W, \lambda)$–ES.

3.2.1 Concept and algorithm

The $(\mu/\mu_W, \lambda)$–ES is a class of ES's which evolves a single candidate solution. At the k-th generation, the new offspring $y_{k+1}^1, \ldots, y_{k+1}^\lambda$ are generated around a weighted mean x_k of the previous parents (candidate solution). The symbol "$/\mu_W$" in $(\mu/\mu_W, \lambda)$–ES specifies that μ parents are 'recombined' into a weighted mean. The parents are selected as the μ best offspring of the previous iteration in terms of the objective function value. The mutation operator of the new offspring points is done by $y_{k+1}^i = x_k + \sigma_k^{\mathrm{ES}} d_k^i$, $i = 1, \ldots, \lambda$,

where d_k^i is drawn from a certain distribution $\mathcal{C}_k$ and σ_k^{ES} is a chosen step size. The weights used to compute the means belong to the simplex set $S = \{(\omega^1, \ldots, \omega^\mu) \in \mathbb{R}^\mu : \sum_{i=1}^{\mu} w^i = 1, w^i \geq 0, i = 1, \ldots, \mu\}$.

The $(\mu/\mu_W, \lambda)$–ES adapts the sampling distribution to the landscape of the objective function. An adaptation mechanism for the step size parameter is also possible. The latter one increases or decreases depending on the landscape of the objective function. Figure 3.4 depicts a 2-dimensional illustration, where one starting from an isotropic distribution is able to adapt its evolution to the landscape of the objective function.



(a) The first generation. (b) The second generation. (c) The third generation.

FIGURE 3.4: A 2-D illustration of an evolution strategy. Generation after generation the sampling distribution and the step size are getting adapted to the landscape of the objective function. The ellipses show the level sets of the objective function.

The algorithmic description of such class of ES then can be given as follows:

3.2.2 Some existing convergence results

For almost three decades, many theoretical works on evolution strategies have focused on convergence toward optima but under very mild assumptions either on the objective functions or on the endogenous strategy parameters [20, 24, 26, 30, 33, 75, 96, 97, 100, 151, 169]. For the objective functions, the sphere problem is among the most frequently studied case [20, 30, 33, 100, 151, 169]. Such problem may seem simple, but the convergence theory behind is rather not trivial [20, 33, 100].

In addition to the assumption on the objective function, most of the existing global convergence results consider simple schemes of Algorithm 3.2. By global convergence, we mean convergence to a stationary point, with a probability one, regardless the starting point. The most theoretical studied algorithm is known as $(1, \lambda)$-ES where the new parent is defined as the best offspring (see the reference [169] and the references therein). The first convergence results were mainly obtained using martingale theory tools [30, 151].

Algorithm 3.2: A general framework for $(\mu/\mu_W, \lambda)$–ES.

Initialization: Choose positive integers λ and μ such that $\lambda \geq \mu$. Choose an initial x_0, an initial step length $\sigma_0^{\text{ES}} > 0$, an initial distribution $\mathcal{C}_0$, and initial weights $(\omega_0^1, \ldots, \omega_0^\mu) \in S$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. **Offspring Generation:** Compute new sample points $Y_{k+1} = \{y_{k+1}^1, \ldots, y_{k+1}^\lambda\}$ such that
$$y_{k+1}^i \;=\; x_k + \sigma_k^{\text{ES}} d_k^i,$$
where d_k^i is drawn from the distribution $\mathcal{C}_k$, $i = 1, \ldots, \lambda$.

2. **Parent Selection:** Evaluate $f(y_{k+1}^i)$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\lambda\}$ by increasing order: $f(\tilde{y}_{k+1}^1) \leq \cdots \leq f(\tilde{y}_{k+1}^\lambda)$.
Select the new parents as the best μ offspring sample points $\{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\mu\}$, and compute their weighted mean
$$x_{k+1} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i.$$

3. **Updates:** Update the step length σ_{k+1}^{ES}, the distribution $\mathcal{C}_{k+1}$, and the weights $(\omega_{k+1}^1, \ldots, \omega_{k+1}^\mu) \in S$. Increment k and return to Step 1.

More recent convergence proofs are based on Markov chains theory, Bienvenüe and François [33] and later Auger [20] proved convergence results for $(1, \lambda)$-SA-ES[1] on the sphere function. The first authors [33] showed that the convergence, or divergence, is conditioned by the ability to prove some recurrence properties of a specific Markov chain. Auger [20] proves sufficient conditions to ensure asymptotic log-linear convergence or divergence of $(1, \lambda)$-SA-ES algorithm. By log-linear convergence, we mean convergence of $1/k \ln(\|x_k\|)$, where x_k is the parent at the generation k.

For non-convex objective functions and using measure theory, Greenwood and Zhu [75] proposed a globally convergent version of $(1, \lambda)$-ES. A self-adaptation that uses 1/5-success rule was incorporated in $(1, \lambda)$-ES, meaning that depending on the percentage of success mutations P_s (i.e. individuals that have better objective function values compared to their parent) recorded over a certain number of generations. The mutation strength (i.e. the step size) is increased after a certain number of generations, if $P_s > 1/5$, and decreased otherwise.

For spherical objective functions[2], Jebalia and Auger [100] prove log-linear convergence

[1] SA stands for Self-Adaptive

[2] f is said to be a spherical function if there exists a strictly increasing function g such as $\forall x \in \mathbb{R}^n f(x) = g(\|x\|)$.

of Algorithm 3.2 in the isotropic case (using an isotropic mutation) and under a scale-invariant adaptation rule (i.e. for a given generation k one has $\sigma_k^{\text{ES}} = \sigma\|x_k\|$ where $\sigma > 0$).

3.2.3 CMA-ES a state of the art for ES

The Covariance Matrix Adaptation - Evolution Strategy (CMA-ES) [85, 86] is regarded as one of the most relevant instances of $(\mu/\mu_W, \lambda)$–ES emphasized in Algorithm 3.2. The success of such method has many reasons. In fact, CMA-ES adapts both the sampling distribution as well as the step size parameter to the landscape of the objective function. One decreases or increases the exploration depending on the landscape of the objective function. CMA-ES is also known to be invariant upon monotonic transformations of the objective function; these transformations are preserving the ranking of the solution (i.e. selection mechanism) which is regarded as a robustness property of CMA-ES [69].

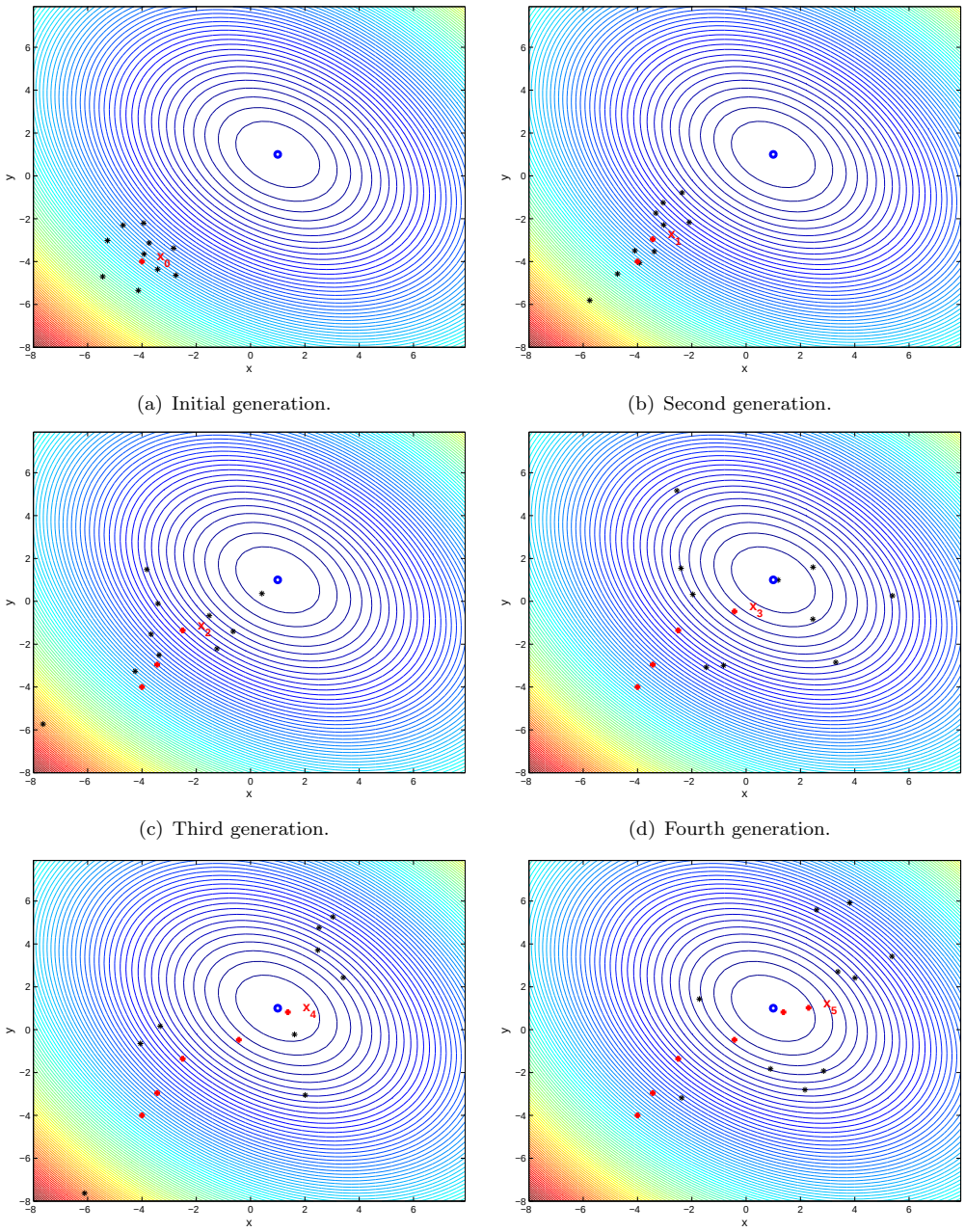
Figure 3.5 depicts the first six generations of CMA-ES on a convex problem. Starting from an isotropic variance, the offspring population is getting adapted to the landscape of the objective function. The secret behind such adaptation processes will be outlined in the rest of Section 3.2.3. Starting from Algorithm 3.2 at a given generation k, we will now describe how the distribution $\mathcal{C}_k$ as well as the step size $\sigma_k^{\text{CMA-ES}}$ are updated in the CMA-ES framework.

3.2.3.1 The parent update

In Algorithm 3.2, the directions d_k^i used for the offspring generation are generally drawn from a given distribution $\mathcal{C}_k$. In CMA-ES context, the distribution is chosen to be a multivariate normal distribution of mean zero and covariance matrix C_k denoted by $\mathcal{N}(0, C_k)$. Following such choice for the mutations distribution, one fulfilled all the mutations requirements specified earlier in Section 3.1.4. The offspring generation is then completed as follows:

$$y_{k+1}^i \;=\; x_k + \sigma_k^{\text{CMA-ES}}\mathcal{N}(0, C_k) \quad \text{, for } i = 1, \ldots, \lambda$$

The covariance matrix C_k reflects the landscape of the objective function, and serves to steer the exploration to better zones. The step size σ_k^{ES} is used as a global scaling factor for the covariance matrix. More insights on both the covariance matrix and the step size parameter will be outlined later in Sections 3.2.3.2 and 3.2.3.3. After the generation of λ individuals, the mean parent is updated using the μ best individuals in terms of the



(a) Initial generation.

(b) Second generation.

(c) Third generation.

(d) Fourth generation.

(e) Fifth generation.

(f) Sixth generation.

FIGURE 3.5: A graphical representation of a 2-dimensional run of CMA-ES where $x_0 = [-4, -4]$, the initial step size $\sigma_0^{\text{CMA-ES}} = 1$, and the covariance matrix is isotropic (i.e. $C_0 = I_2$). The population size is $\lambda = 10$, the new parent is chosen using the $\mu = 5$ best individuals. The ellipses show the level sets of the objective function $f(x) = (x_1 + x_2 - 2)^2 + (x_1 - x_2)^2$. The optimum is located at the point $[1, 1]$.

objective function:

$$x_{k+1} = \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i = x_k + \sigma_k^{\text{CMA-ES}} \sum_{i=1}^{\mu} \omega_i \tilde{d}_{k+1}^i,$$

where $\tilde{y}^i_{k+1}$ (resp. $\tilde{d}^i_{k+1}$) is the the i^{th} best individual (resp. direction) out of the offspring generation. The number μ is chosen to be equal to $\frac{\lambda}{2}$ and the weights $(\omega_i)_{1\leq i\leq \mu}$ are normalized, i.e. satisfying $\sum_{i=1}^{\mu}\omega_i = 1$. The default weights are defined as follows:

$$\omega_i \;=\; \frac{\ln(\mu+2)-\ln(i)}{\mu\ln(\mu+2)-\ln(\mu!)}, \quad \text{for } i=1,\ldots,\mu.$$

3.2.3.2 Covariance matrix update

The adaptation of the covariance matrix targets to include second order information of the underlying objective function (similarly to the inverse Hessian matrix approximation in the Quasi-Newton method in classical optimization) [130]. The update of the covariance matrix is based on two update terms: the rank-one update term [85] and the rank-μ update term [82]. The first one is computed using the so-called evolution path $p^c_k \in \mathbb{R}^n$, updated iteratively as

$$p^c_{k+1} \;=\; (1-c_c)p^c_k + [c_c(2-c_c)\mu_f]^{\frac{1}{2}}(x_{k+1}-x_k)/\sigma^{\text{CMA-ES}}_k,$$

where $c_c \in (0,1]$ is a positive constant depending on the problem dimension n, the quantity $\mu_f = 1/\sum_{i=1}^{\mu}\omega_i^2$ is a measure characterizing the considered recombination, and is known as the variance effective selection mass.

The evolution path reflects the steps followed by the mean parent, the rank-one update consists in adding to the covariance matrix a term that geometrically deforms the density in the direction p^c_{k+1} (the next generation is more likely sampled in the direction of p^c_{k+1}, such statement is equivalent to adding the term $(p^c_{k+1})(p^c_{k+1})^\top$ to the covariance matrix). The rank-mu update term is composed of the rank-mu matrix $\sum_{i=1}^{\mu}\omega_i\,(\tilde{d}^i_k)(\tilde{d}^i_k)^\top$, such update turns out to conduct a natural gradient[3] update of the distribution parameters [13]. Thus, CMA-ES updates the covariance matrix of C_k as follows:

$$C_{k+1} \;=\; (1-c_1-c_\mu)\,C_k + c_1(p^c_{k+1})(p^c_{k+1})^\top + c_\mu\sum_{i=1}^{\mu}\omega_i\,(\tilde{d}^i_k)(\tilde{d}^i_k)^\top.$$

The initial evolution path p^c_0, c_c, c_1, and c_μ are the algorithm parameters (see [78] for the default values).

[3]The natural gradient is defined as the gradient of $J(\theta)$ the expected objective function under a search distribution $p(x/\theta)$, namely $J(\theta) = E(f(x)/\theta) = \int f(x)p(x/\theta)$.

3.2.3.3 Step size update

The CMA-ES's step size is adapted iteratively according to:

$$\sigma_{k+1}^{\text{CMA-ES}} \;=\; \sigma_k^{\text{CMA-ES}} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|p_{k+1}^\sigma\|}{E\|\mathcal{N}(0,I)\|} - 1\right)\right),$$

where $E\|\mathcal{N}(0,I)\| = \sqrt{2}\Gamma(\frac{n+1}{2})/\Gamma(\frac{n}{2})$[4] is the expectation of the ℓ_2 norm of an $\mathcal{N}(0,I)$ distributed random vector, the constants c_σ, d_σ are positive constants, and $p_{k+1}^\sigma \in \mathbb{R}^n$ is the current state of the so-called conjugate evolution path [84]. The latter one is updated using a rank-one update multiplied by the covariance matrix inverse square root of the last generation, i.e. meaning $[C_k]^{-\frac{1}{2}}(x_{k+1}-x_k)/\sigma_k^{\text{CMA-ES}}$. The complete update formula is as follows:

$$p_{k+1}^\sigma \;=\; (1-c_\sigma)p_k^\sigma + \sqrt{c_\sigma(2-c_\sigma)\mu_f}[C_k]^{-\frac{1}{2}}(x_{k+1}-x_k)/\sigma_k^{\text{CMA-ES}},$$

the constants p_0^c, c_σ, and d_σ are parameters of the algorithm (see [78] for the default values).

3.2.4 Local meta-models and ES's

The main difficulty for applying ES's to real-world applications is that ES's may need a large number of objective function evaluations to converge. Moreover, the objective function evaluations are not always cheap in terms of CPU cost in many real-world applications. Either an explicit objective function may not be available, or its evaluation can be computationally very expensive. In all cases, it is necessary to estimate the objective function using model based techniques, known as fitness approximation in the evolutionary computation community. For ES's, various model based technics have been proposed. Jin [101] presents a comprehensive survey of the most popular model based technics currently used with evolutionary algorithms, in particular, evolution strategies.

$(\mu/\mu_W, \lambda)$–ES does not use explicitly information from the objective function except for the ranking. Thus, a model that can preserve the ranking of the objective function would be enough. On the light of such idea, Kern et al [105] proposed an algorithm where the quality of a meta-model is measured using only the information coming from the change in the exact ranking of the best individuals. The construction of the meta-model is based on a locally weighted regression assisted by an approximate ranking procedure [147].

[4] $\Gamma(.)$ denote the Gamma function, i.e. $\Gamma(t) = \int_0^{+\infty} x^{t-1}e^{-x}\mathrm{d}x$.

3.2.4.1 Locally weighted regression

Earlier in Section 2.1.2 of Chapter 2, we emphasized approximation model technics based on a second-order Taylor series expansion. Basically, in a derivative-free context, one uses quadratic interpolation to build the model. Based on the same idea, locally weighted regression [14] attempts to build a model using an interpolation set (known as training data in the evolutionary computing community). Thanks to a kernel weighting procedure, the constructed models tend to be more adapted to the topography of the objective function. An algorithmic description can be made as follows. Let $x \in \mathbb{R}^n$ be the point to be evaluated with an approximate interpolation model m. Let $Y = \{y^i\}_{1 \leq i \leq p}$ be an interpolation set of p points near the query point x and $\{f(y^i)\}_{1 \leq i \leq p}$ the associated objective function values. The local model for a given interpolation set at the point x is of the form:

$$m(x, \alpha_\phi) \quad = \quad \sum_{j=1}^{q} \alpha_j \phi_j(x),$$

where $\alpha_\phi = (\alpha_1, \ldots, \alpha_q)^\top \in \mathbb{R}^q$ and $\{\phi_i\}_{i=1}^q$ be a given basis of $\mathcal{P}_n^d$, which is a set of q polynomials of degree $\leq d$ (see Section 2.1.2 of Chapter 2 for more details).

Rather than minimizing directly the gap between the model values $\{m(y^i, \phi)\}_{1 \leq i \leq p}$ and the interpolation values $\{f(y^i)\}_{1 \leq i \leq p}$ to find the best coefficients α_ϕ, locally weighted regression minimizes the same gap but by mostly taking into account more the closest points. The procedure is equivalent to minimizing a training criterion function C with respect to the interpolation coefficients α_ϕ of the local model m at the point x. The criterion function has the following form:

$$C(x) = \sum_{j=1}^{p} \left[(m(y^j, \alpha_\phi) - y^j)^2 K\left(\frac{d(y^j, x)}{h} \right) \right], \tag{3.10}$$

where $K(.)$ is a kernel weighting function, $d(y^j, x)$ is the distance between the interpolation point y^j and x, and h is a bandwidth chosen as the distance of k-th nearest neighbor interpolation point, in Y, to the point x. The distance used in [105] for $d(y^j, x)$ is preconditioned with the covariance matrix C used in the CMA-ES, i.e. $d(y^j, x) = \|y^j - x\|_C = \sqrt{(y^j - x)^\top C^{-1} (y^j - x)}$. The reason behind such a choice is that the covariance matrix contains information on the local topography of the objective function that one is trying to exploit [105]. A bi-quadratic form is generally used as a kernel function:

$$K(\xi) = \begin{cases} (1 - \xi^2)^2 & \text{if} \quad \xi < 1, \\ 0 & \text{otherwise.} \end{cases}$$

Local regression models are shown to be not very dependent to the choice of the kernel function [105], and are not used until sufficiently many objective function evaluations have been stored.

The minimization of C turns to be equivalent to solve the following normal equations:

$$\Big((WM(\phi,Y))^\top WM(\phi,Y)\Big)\alpha_\phi \;=\; (WM(\phi,Y))^\top Wf(Y), \tag{3.11}$$

where $M(\phi,Y)$ is the coefficient matrix, $f(Y) = (f(y^1), f(y^2), \ldots, f(y^p)^\top$, and $W = \text{diag}\left(\sqrt{K(d(y^1,x)/h)}, \sqrt{K(d(y^2,x)/h)}, \ldots, \sqrt{K(d(y^p,x)/h)}\right)$.

3.2.4.2 Approximate ranking procedure

Using the locally weighted regression a local model is built, to incorporate such model in the ES an approximate ranking procedure is needed [147]. As our ES algorithm uses only the ranking information from the objective function, the quality of the model will be measured on how our built model is representing the true ranking. The ranking procedure aims to tell if the model is good enough to exploit or new true objective function evaluations should be performed. In the CMA-ES framework, the resulting method is called the local-meta-model CMA-ES (lmm-CMA-ES) [105]. The choice of the model is based on the idea that one adds to the interpolation set points until the rank of the points remains unchanged for two consecutive iteration cycles. As the ranking process for ES's depends only on the μ best points, the ranking invariance is checked only for the μ best individuals, this means that the predicted ranking in the μ first position should not change for two consecutive iterations to accept the model. A detailed description of the approximation ranking procedure is depicted in Algorithm 3.3.

For the first call to the approximate ranking procedure, the number n_{init} is initialized with the value λ, and gets adapted afterward for the next calls. For each iteration of the procedure, the objective function is evaluated on a batch number n_b of points until the evaluation rank of the μ best individuals, based on the model, is kept unchanged for two consecutive iterations. The number n_b is chosen to be equal to $\max(1, \lambda/10)$. To construct a good model, the ranking procedure ends up with $n_{\text{init}} + i * n_b$ individual evaluations, where $i \in \{1, \ldots, (\lambda - n_{\text{init}})/n_b\}$ represents the number of iterations needed to get the model accepted.

Algorithm 3.3: Approximate ranking procedure.

1. **Building a model:** build of model m and evaluate the points $m(y^k), k = 1, \ldots, \lambda$ based on an interpolation set Y.

2. **Ranking:** rank individuals according to m, let $R_0^\mu = \{\tilde{y}^1, \ldots, \tilde{y}^\mu\}$ by increasing order: $m(\tilde{y}^1) \leq \cdots \leq m(\tilde{y}^\lambda)$.

3. **Evaluating :** Evaluate the individuals $\{\tilde{y}^i\}_{1 \leq i \leq n_{\text{init}}}$ using the objective function, and add their evaluations to the set Y.

4. **For** $i \in \{1, \ldots, (\lambda - n_{\text{init}})/n_b\}$,

   - Build a model m based on the point set Y, and evaluate the points $m(y^k), k = 1, \ldots, \lambda$.
   - Rank individuals according to m, let $R_k^\mu = \{\tilde{y}^1, \ldots, \tilde{y}^\mu\}$ by increasing order: $m(\tilde{y}^1) \leq \cdots \leq m(\tilde{y}^\lambda)$.
   - If $R_k^\mu = R_{k-1}^\mu$, the model m is accepted and we exit from the loop.
   - If $R_k^\mu \neq R_{k-1}^\mu$, evaluate the best n_{init} (not yet evaluated) using the objective function, and add their evaluations to the set Y.

5. If $i > 2$ then set n_{init} to $\min(n_{\text{init}}, \lambda - n_b)$, otherwise if $i < 2$ then set n_{init} to $\max(n_{\text{init}} - n_b, n_b)$.

3.3 Conclusion

In this chapter, we presented an overview of ES's and explained their philosophy and mechanisms, a detailed description can be found in [30, 32, 32, 142, 150]. We describe succinctly a class of ES's, denoted by $(\mu/\mu_W, \lambda)$–ES, for which we cited some theoretical aspects, in particular, the main existing global convergence properties of ES algorithms [20, 24, 26, 30, 33, 75, 96, 97, 100, 151, 169]. We closed the chapter by first given a detailed description of CMA-ES [85, 86], and then explaining how quadratic models were used in a large class of ES's.

This chapter was introduced in preparation to what comes next. The next chapter will detail our first contribution of this thesis, where we show how to equip $(\mu/\mu_W, \lambda)$–ES with some direct search techniques (introduced in Chapter 2) to rigorously achieve a form of global convergence under reasonable assumptions. Later, we will explicit another way to incorporate surrogate quadratic models in our proposed ES to enhance the performance without deteriorating the global convergence properties of the proposed algorithm.

In all our numerical experiments, we choose CMA-ES as our evolutionary strategy, on top of which we tested all the proposed modifications.

Chapter 4

Globally Convergent Evolution Strategies

In this chapter, we emphasize the first contribution of this thesis [58]. We show how to modify $(\mu/\mu_W, \lambda)$–ES to rigorously achieve a form of global convergence under reasonable assumptions.

As far as we know (see Section 3.2.2), most existing global convergence results focused on specific objective functions where the most studied one is the sphere problem [20, 30, 33, 100, 151, 169]. Other existing global convergence results consider a weak framework of $(\mu/\mu_W, \lambda)$–ES, particularly $(1, \lambda)$-ES [30, 75, 151, 169]. Previously mentioned works do not take into account recombination (Section 3.1.2). Recent studies start to include the recombination constraint for some specific problems and with strong assumptions. For instance, asymptotic results for $(\mu/\mu_W, \lambda)$–ES are proved for spherical functions in the isotropic case and under a scale-invariant adaptation rule for the step size [100].

In our framework, we consider the algorithm $(\mu/\mu_W, \lambda)$–ES as general as possible, in the sense that no assumptions are made on the generation distribution. Meanwhile, one needs to assume the density of certain limit directions in the unit sphere. The modification of $(\mu/\mu_W, \lambda)$–ES consists essentially of the reduction of the size of the steps whenever a sufficient decrease condition on the function values is not verified. By a sufficient decrease condition we mean a decrease of the type $f(x_{k+1}) \leq f(x_k) - \rho(\sigma_k)$, where σ_k stands for the step size parameter and $\rho(\cdot)$ obeys some properties, in particular $\rho(t)/t \to 0$ when $t \downarrow 0$ (see Section 2.2.2). When such a condition is satisfied, the step size can be reset to the one designed by the ES itself, as long as this latter one is sufficiently large. We suggest three ways of imposing sufficient decrease for which global convergence holds under reasonable assumptions.

The technique that we use to proove the global convergence of a such ES resembles what is done in direct search [18, 52, 166]. In particular, given the type of random sampling used in these ES, our work is inspired by direct-search methods for nonsmooth functions outlined in Section 2.2.2, where one must use a set of directions asymptotically dense in the unit sphere and with a sufficient decrease condition to control the step size. One way of imposing such condition in the type of ES under consideration is to apply it directly to the sequence of weighted means. However, ES are population-based algorithms where a sample set of the offspring is generated at every iteration. Other forms of imposing this type of decrease which involve the maximum value of the best offspring are also found globally convergent. In fact, requiring a sufficient decrease on the sequence of maximum best offspring values renders a globally convergent algorithm. Furthermore, we will show that demanding this maximum value to sufficiently decrease the weighted mean leads also to global convergence.

The approach we have taken in our thesis is (i) to focus on deterministic objective functions and (ii) to analyze each algorithm deterministically (considering a single realization of a stochastic algorithm). In such a way, we were able to use the Clarke calculus and avoided imposing additional assumptions on the objective function.

The chapter is organized as follows. In Section 4.1, we first describe how to modify such algorithms to enable them for global convergence. The second part is devoted to the analysis of global convergence of the modified ES versions. Our numerical experiments comparing the different modified versions of CMA-ES are described in Section 4.2. Finally, in Section 4.3, we draw some conclusions and perspectives.

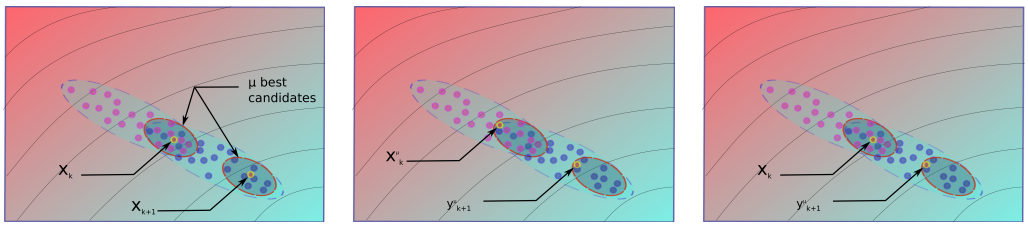
4.1 A class of evolution strategies provably global convergent

4.1.1 Globally convergent evolution strategies

The main question we address in this chapter is how to change $(\mu/\mu_W, \lambda)$–ES algorithm (see Algorithm 3.2 in Chapter 3), in a minimal way, to make it enjoy some convergence properties, while preserving as much as possible the original design and goals. We will target at global convergence in the sense of nonlinear optimization, in other words we would like to prove some limit form of stationarity for any output sequence of iterates generated by the algorithm (i.e., for any realization of the algorithm), and we would like to do this independently of the starting point.

The modifications to the algorithm will be essentially two, and they have been widely used in the field of nonlinear optimization, with and without derivatives. First we need to control the size of the steps taken, and thus we will update separately a step size parameter σ_k, letting it take the value of σ_k^{ES} whenever possible, where σ_k^{ES} is the original step size of the considered evolution strategy. Controlling the step size is essential as we know that most steps used in nonlinear optimization are too large away from stationarity — an example is Newton's method without a line search, which may take arbitrarily large steps if not started sufficiently close to a problem solution. Secondly we need to impose some form of sufficient decrease on the objective function values to be able to declare an iteration successful and thus avoiding a step size reduction. These two techniques, step size update and imposition of sufficient decrease on the objective function values, are thus closely related since an iteration is declared unsuccessful and the step size reduced when the sufficient decrease condition is not satisfied. This condition involves a function $\rho(\sigma_k)$ of the step size σ_k, where $\rho(\cdot)$ is a forcing function [108] (see Definition 2.8, one can think for instance of $\rho(t) = t^2$).

Since the $(\mu/\mu_W, \lambda)$–ES algorithm evaluates the objective function at the offspring sample points but then computes new points around a weighted sum of the parents selected, it is not clear how this does impose sufficient decrease. In fact, there are several ways of proceeding in the follwing. A first possibility (denoted by mean/mean) is to require the weighted means to sufficiently decrease the objective function, see Figure (4.1(a)) below, which obviously requires an extra function evaluation per iteration.



(a) The mean/mean version. (b) The max/max version. (c) The max/mean version.

FIGURE 4.1: A 2-D illustration of three possible globally convergent evolution strategies. The ellipses show the level sets of the objective function.

A second possibility to impose sufficient decrease (referred to as max/max), based entirely on the objective function values already computed for the parent samples, is to require the maximum of these values to be sufficiently decreasing, see Figure (4.1(b)). Then, it would immediately occur to combine these first two possibilities, asking the new maximum value to reduce sufficiently the value of the previous mean or, vice-versa, requiring the value of the new mean to reduce sufficiently the previous maximum. The lack of theoretical support of the latter possibility made us consider only the first one,

called max/mean, see the Figure (4.1(c)). Algorithm 4.1 outlines the modified form of the $(\mu/\mu_W, \lambda)$–ES.

The version mean/mean is clear in the sense that it imposes the sufficient decrease condition directly on the function values computed at the sequence of minimizer candidates, the weighted sums. It is also around these weighted sums that new points are randomly generated. Versions max/max and mean/max, however, operate based or partially based on the function values at the parents samples (on the maximum of those). Thus, in these two versions, one needs to impose a condition of the form (4.1) below to balance the function values at the parents samples and the function value at the weighted sum.

When the objective function is convex, condition (4.1) would be true for any weights in S, but neither such a condition is realistic when optimizing without derivatives nor would perhaps the type of techniques explored in this work be the most appropriate under such a scenario. Note that one also imposes bounds on all directions d_k^i used by the algorithm. This modification is, however, very mild since the lower bound $d_{\min}$ can be chosen very close to zero and the upper bound $d_{\max}$ set to a very large number. Moreover, one can think of working always with normalized directions which removes the need to impose such bounds.

4.1.2 Convergence

Under appropriate assumptions we will now prove global convergence of the modified versions of the considered class of ES (again, by global convergence, we mean some form of limit first-order stationary for arbitrary starting points). Our convergence analysis is inspired by direct-search methods for nonsmooth functions outlined in Section 2.2. The analysis of the algorithm is done deterministically, as if we were considering a single realization of a stochastic algorithm.

4.1.2.1 The step size behavior

As we have seen before, an iteration is considered successful only if it produces a point that has sufficiently decreased some value of f. Insisting on a sufficient decrease will guarantee that a subsequence of step sizes will converge to zero. In fact, since $\rho(\sigma_k)$ is a monotonically nondecreasing function of the step size σ_k, we will see that such a step size cannot be bounded away from zero since otherwise some value of f would tend to $-\infty$. Imposing sufficient decrease will make it harder to have a successful step and therefore will generate more unsuccessful steps. We start thus by showing that there is a subsequence of iterations for which the step size parameter σ_k tends to zero.

Algorithm 4.1: A class of globally convergent ES's.

Initialization: Use the same initialization of Algorithm 3.2. Choose constants $\beta_1, \beta_2, d_{\min}, d_{\max}$ such that $0 < \beta_1 \leq \beta_2 < 1$ and $0 < d_{\min} < d_{\max}$. Select a forcing function $\rho(\cdot)$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. **Offspring Generation:** Compute new sample points $Y_{k+1} = \{y_{k+1}^1, \ldots, y_{k+1}^\lambda\}$ such that
$$y_{k+1}^i \;=\; x_k + \sigma_k d_k^i,$$
where d_k^i is drawn from the distribution $\mathcal{C}_k$ and obeys $d_{\min} \leq \|d_k^i\| \leq d_{\max}$, $i = 1, \ldots, \lambda$.

2. **Parent Selection:** Evaluate $f(y_{k+1}^i)$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\lambda\}$ by increasing order: $f(\tilde{y}_{k+1}^1) \leq \cdots \leq f(\tilde{y}_{k+1}^\lambda)$.
Select the new parents as the best μ offspring sample points $\{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\mu\}$, and compute their weighted mean
$$x_{k+1}^{trial} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i.$$
Evaluate $f(x_{k+1}^{trial})$. In versions max/max and max/mean, update the weights, if necessary, such that $(\omega_k^1, \ldots, \omega_k^\mu) \in S$ and
$$f(x_{k+1}^{trial}) \;=\; f\left(\sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i\right) \;\leq\; \sum_{i=1}^{\mu} \omega_k^i f(\tilde{y}_{k+1}^i). \tag{4.1}$$

3. **Imposing Sufficient Decrease:**
If (version mean/mean)
$$f(x_{k+1}^{trial}) \;\leq\; f(x_k) - \rho(\sigma_k), \tag{4.2}$$
or (version max/max)
$$f(\tilde{y}_{k+1}^\mu) \;\leq\; f(x_k^\mu) - \rho(\sigma_k), \tag{4.3}$$
or (version max/mean)
$$f(\tilde{y}_{k+1}^\mu) \;\leq\; f(x_k) - \rho(\sigma_k), \tag{4.4}$$
then consider the iteration successful, set $x_{k+1} = x_{k+1}^{trial}$, and $\sigma_{k+1} \geq \sigma_k$ (for example $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\mathrm{ES}}\}$). Set $x_{k+1}^\mu = \tilde{y}_{k+1}^\mu$ in version max/max.
Otherwise, consider the iteration unsuccessful, set $x_{k+1} = x_k$ (and $x_{k+1}^\mu = x_k^\mu$ for max/max) and $\sigma_{k+1} = \bar{\beta}_k \sigma_k$, with $\bar{\beta}_k \in (\beta_1, \beta_2)$.

4. **ES Updates:** Update the ES step length $\sigma_{k+1}^{\mathrm{ES}}$, the distribution $\mathcal{C}_k$, and the weights $(\omega_{k+1}^1, \ldots, \omega_{k+1}^\mu) \in S$. Increment k and return to Step 1.

Lemma 4.1. *Consider a sequence of iterations generated by Algorithm 4.1 without any stopping criterion. Let f be bounded below. Then* $\liminf_{k\to+\infty} \sigma_k = 0$.

Proof. Suppose that there exists a $\sigma > 0$ such that $\sigma_k > \sigma$ for all k. If there is an infinite number of successful iterations, this leads to a contradiction to the fact that f is bounded below. Since ρ is a nondecreasing, positive function, one has $\rho(\sigma_k) \geq \rho(\sigma) > 0$. Let us consider the three versions separately, as we shall see now.

In the version mean/mean, we obtain $f(x_{k+1}) \leq f(x_k) - \rho(\sigma)$ for all k, which obviously contradicts the boundedness below of f. In the version max/max, we obtain $f(x^{\mu}_{k+1}) \leq f(x^{\mu}_k) - \rho(\sigma)$ for all k, which also trivially contradicts the boundedness below of f. For the max/mean version, one has

$$f(\tilde{y}^i_{k+1}) \;\leq\; f(x^{\mu}_{k+1}) \;\leq\; f(x_k) - \rho(\sigma_k), \quad i = 1, \ldots, \mu.$$

Thus, multiplying these inequalities by the weights ω^i_k, $i = 1, \ldots, \mu$, and adding them up, lead us to

$$\sum_{i=1}^{\mu} \omega^i_k f(\tilde{y}^i_{k+1}) \;\leq\; f(x_k) - \rho(\sigma_k),$$

and from condition (4.1) imposed on the weights in Step 2 of Algorithm 4.1, we obtain $f(x_{k+1}) \leq f(x_k) - \rho(\sigma_k)$, and the contradiction is also easily reached.

The proof is thus completed if there is an infinite number of successful iterations. However, if no more successful iterations occur after a certain order, then this also leads to a contradiction. The conclusion is that one must have a subsequence of iterations driving σ_k to zero. □

From the fact that σ_k is only reduced in unsuccessful iterations and by a factor not approaching zero, one can then conclude the following.

Lemma 4.2. *Consider a sequence of iterations generated by Algorithm 4.1 without any stopping criterion. Let f be bounded below.*

There exists a subsequence K of unsuccessful iterates for which $\lim_{k\in K} \sigma_k = 0$.

If the sequence $\{x_k\}$ is bounded, then there exists an x_* and a subsequence K of unsuccessful iterates for which $\lim_{k\in K} \sigma_k = 0$ *and* $\lim_{k\in K} x_k = x_*$.

Proof. From Lemma 4.1, there must exist an infinite subsequence K of unsuccessful iterates for which σ_{k+1} goes to zero. In a such case we have $\sigma_k = (1/\bar{\beta}_k)\sigma_{k+1}$, $\bar{\beta}_k \in (\beta_1, \beta_2)$, and $\beta_1 > 0$, and thus $\sigma_k \to 0$, for $k \in K$, too.

The second part of the lemma is also easily proved by extracting a convergent subsequence of the subsequence K of the first part for which x_k converges to x_*. □

The above lemma ensures under mild conditions the existence of convergent subsequences of unsuccessful iterations for which the step size tends to zero. Known as refining subsequences (see Section 2.11).

4.1.2.2 Global convergence

The global convergence in our case is extracted from refining subsequences. One will assume that the function f is Lipschitz continuous near the limit point x_* of a refining subsequence, so that the Clarke generalized derivative [43]

$$f^\circ(x_*; d) \;=\; \limsup_{x \to x_*, t \downarrow 0} \frac{f(x+td) - f(x)}{t}$$

exists for all $d \in \mathbb{R}^n$. The point x_* is then Clarke stationary if $f^\circ(x_*; d) \geq 0$, $\forall d \in \mathbb{R}^n$ (See Section 2.2.3.2 for more details on the non-smooth Clarke calculus).

Our first global convergence result concerns only the mean/mean version.

Theorem 4.3. *Consider the version mean/mean and let* $a_k = \sum_{i=1}^{\mu} \omega_k^i d_k^i$. *Assume that the directions* d_k^i*'s and the weights* ω_k^i*'s are such that* $\|a_k\|$ *is bounded away from zero when* $\sigma_k \to 0$. *Let* x_* *be the limit point of a convergent subsequence of unsuccessful iterates* $\{x_k\}_K$ *for which* $\lim_{k \in K} \sigma_k = 0$. *Assume that* f *is Lipschitz continuous near* x_* *with constant* $\nu > 0$.

If d *is a limit point of* $\{a_k/\|a_k\|\}_K$, *then* $f^\circ(x_*; d) \geq 0$.

If the set of limit points $\{a_k/\|a_k\|\}_K$ *is dense in the unit sphere, then* x_* *is a Clarke stationary point.*

Proof. Let d be a limit point of $\{a_k/\|a_k\|\}_K$. Then it must exist a subsequence of K' of K such that $a_k/\|a_k\| \to d$ on K'. On the other hand, we have for all k that

$$x_{k+1}^{trial} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i \;=\; x_k + \sigma_k \sum_{i=1}^{\mu} \omega_k^i d_k^i \;=\; x_k + \sigma_k a_k,$$

and, for $k \in K$,

$$f(x_k + \sigma_k a_k) \;>\; f(x_k) - \rho(\sigma_k).$$

Also, since the directions d_k^i and the weights are bounded above for all k and i, a_k is bounded above for all k, and so $\sigma_k \|a_k\|$ tends to zero when σ_k does.

Thus, from the definition of the Clarke generalized derivative,

$$\begin{aligned} f^\circ(x_*;d) &= \limsup_{x\to x_*,t\downarrow 0} \frac{f(x+td)-f(x)}{t} \\ &\geq \limsup_{k\in K'} \frac{f(x_k+\sigma_k\|a_k\|(a_k/\|a_k\|))-f(x_k)}{\sigma_k\|a_k\|} - r_k, \end{aligned}$$

where, from the Lipschitz continuity of f near x_*,

$$r_k \;=\; \frac{f(x_k+\sigma_k a_k)-f(x_k+\sigma_k\|a_k\|d)}{\sigma_k\|a_k\|} \;\leq\; \nu\left\|\frac{a_k}{\|a_k\|}-d\right\|$$

tends to zero on K'. Finally, since $\|a_k\|$ is bounded away from zero in K',

$$\begin{aligned} f^\circ(x_*;d) &\geq \limsup_{k\in K'} \frac{f(x_k+\sigma_k a_k)-f(x_k)+\rho(\sigma_k)}{\sigma_k\|a_k\|} - \frac{\rho(\sigma_k)}{\sigma_k\|a_k\|} - r_k \\ &= \limsup_{k\in K'} \frac{f(x_k+\sigma_k a_k)-f(x_k)+\rho(\sigma_k)}{\sigma_k\|a_k\|} \\ &\geq 0. \end{aligned}$$

Since the Clarke generalized derivative $f^\circ(x_*;\cdot)$ is continuous in its second argument [43], it is then evident that if the set of limit points $\{a_k/\|a_k\|\}_K$ is dense in the unit sphere, $f^\circ(x_*;d)\geq 0$ for all $d\in\mathbb{R}^n$. □

Now we prove global convergence for the two other versions (max/max and max/mean).

Theorem 4.4. *Consider the versions max/max and max/mean. Let x_* be the limit point of a convergent subsequence of unsuccessful iterates $\{x_k\}_K$ for which $\lim_{k\in K}\sigma_k = 0$. Assume that f is Lipschitz continuous near x_* with constant $\nu>0$.*

If d is a limit point of $\{d_k^{i_k}/\|d_k^{i_k}\|\}_K$, where $i_k\in \operatorname{argmax}_{1\leq i\leq\mu} f(y_{k+1}^i)$, then $f^\circ(x_*;d)\geq 0$.

If, for each $i\in\{1,\ldots,\mu\}$, the set of limit points $\{d_k^i/\|d_k^i\|\}_K$ is dense in the unit sphere, then x_* is a Clarke stationary point.

Proof. The proof follows the same lines of the proof of the mean/mean version. In the max/max case, one departs from the inequality that is true when $k\in K$,

$$f(x_{k+1}^\mu) \;>\; f(x_k^\mu)-\rho(\sigma_k),$$

which implies for a certain i_k

$$f(y_{k+1}^{i_k}) \;=\; f(x_{k+1}^\mu) \;>\; f(x_k^\mu)-\rho(\sigma_k).$$

Now, notice that $x^{\mu}_{k+1} = x^{\mu}_{k} = \cdots = x^{\mu}_{k-p_k}$, where $k - p_k - 1$ is the index of the last successful iteration before k. Thus,

$$f(y^{i_k}_{k+1}) > \; f(x^{\mu}_{k-p_k}) - \rho(\sigma_k) \;\geq\; f(\tilde{y}^{i}_{k-p_k}) - \rho(\sigma_k), \quad i = 1, \ldots, \mu.$$

Multiplying these inequalities by the weights $\omega^{i}_{k-p_k-1}$, $i = 1, \ldots, \mu$, and adding them up implies

$$f(y^{i_k}_{k+1}) \;>\; \sum_{i=1}^{\mu} \omega^{i}_{k-p_k-1} f(\tilde{y}^{i}_{k-p_k}) - \rho(\sigma_k),$$

Condition (4.1) imposed on the weights in Step 2 of Algorithm 4.1 with k replaced by $k - p_k - 1$ implies

$$f(y^{i_k}_{k+1}) \;>\; f\left(\sum_{i=1}^{\mu} \omega^{i}_{k-p_k-1} \tilde{y}^{i}_{k-p_k}\right) - \rho(\sigma_k).$$

Since $\sum_{i=1}^{\mu} \omega^{i}_{k-p_k-1} \tilde{y}^{i}_{k-p_k} = x^{trial}_{k-p_k} = x_{k-p_k} = x_k$ (because $k - p_k - 1$ is successful and $k - p_k, \ldots, k$ are unsuccessful) and $y^{i_k}_{k+1} = x_k + \sigma_k d^{i_k}_{k}$, we arrive at

$$f(x_k + \sigma_k d^{i_k}_{k}) \;>\; f(x_k) - \rho(\sigma_k). \tag{4.5}$$

(If there is no successful iteration before the k-th one, then, since $x_0 = x^{\mu}_0$, we will directly obtain (4.5).)

Note that in the max/mean version we arrive directly at $f(x_k + \sigma_k d^{i_k}_{k}) > f(x_k) - \rho(\sigma_k)$.

From this point, and for both cases (max/max and max/mean), the proof is nearly identical to the proof of Theorem 4.3 (in particular note that $d^{i_k}_{k}$ is forced to be bounded away from zero by Algorithm 4.1). □

When f is strict differentiable at x_* (in the sense of Clarke, see Section 2.2.3.2, meaning that there exists $\nabla f(x_*)$ such that $f^{\circ}(x_*; d) = \langle \nabla f(x_*), d \rangle$ for all d), one can conclude that $\nabla f(x_*) = 0$.

4.1.3 Convergence assumptions

Global convergence in Theorems 4.3 and 4.4 is shown under several additional assumptions. The first one is the bounds on the step length $d_{\min} \leq \|d^{i}_{k}\| \leq d_{\max}$, such assumption is quite irrelevant, as in practice for all the tested problems these step lengths were never seen out of the bounds $d_{\min}$ and $d_{\max}$. The boundedness of $\|a_k\|$ away from zero is also not very hard to fulfill, as if one has $a_k = \sum_{i=1}^{\mu} \omega^{i}_{k} d^{i}_{k} = 0$ it suffices to modify the weights $\{\omega^{i}_{k}\}_{1 \leq i \leq \mu}$ so that $a_k \neq 0$.

The assumption that any subsequence of normalized steps is dense on the unit sphere is less trivial. In the sense that the assumption regarding the directions a_k applies to a given refining subsequence K and not to the whole sequence of iterates, but such a strengthening of the requirements on the density of the directions seems necessary for these type of directional methods (see [18, 166]).

Then, the question that arises concerns the density in general of the a_k's in the unit sphere. For the purpose of this discussion, and to keep things simple, let us assume that the weights are fixed for all k (which is a valid choice for Theorem 4.3 but not for Theorem 4.4). Let us assume also that d_k^i's are drawn from a multivariate normal distribution with mean 0 and covariance matrix C. The direction $a_k = \sum_{i=1}^{\mu} \omega^i d_k^i$ is then a realization of a random vector A following a multivariate normal distribution with mean 0 and covariance matrix $\sum_{i=1}^{\mu} (\omega^i)^2 C$. Then, for any $y \in \mathbb{R}^n$ such that $\|y\| = 1$ and for any $\delta \in (0, 1)$, there exists a positive constant η such that

$$P(\cos(A/\|A\|, y) \geq 1 - \delta) \geq \eta \tag{4.6}$$

(see for instance the proof of Lemma B.2 in [73]), such property guarantees us the density of the a_k's in the unit sphere.

Finally, under the random generation framework of the previous paragraph one can also see that we could fix an $M > 0$ (preferably small) at the initialization of the algorithm and then re-sample the d_k^i's again whenever $\|a_k\| < M$. The density of the a_k's in the unit sphere (with probability one) would then result from the fact that, for the same reasons, for any $y \in \mathbb{R}^n$ such that $\|y\| = 1$ and for any $\delta \in (0, 1)$, there would still exist a positive constant η such that $P(\cos(A/\|A\|, y) \geq 1 - \delta, \|A\| \geq M) \geq \eta$.

4.2 Numerical experiments

We made a number of numerical experiments to try to measure the effect of our modifications of ES. We are mainly interested in observing the changes that occur in ES in terms of an efficient and robust search of stationarity. We chose CMA-ES as our evolutionary strategy, on top of which we tested our globally convergent modifications. For CMA-ES details the reader is referred to Section 3.2.3.

For our numerical experiments, we first compare our modifications of CMA-ES among each other and choose the best modified version. For the second part, we have compared the chosen modified CMA-ES and the pure one with the direct search method MADS for which we used the implementation given in the NOMAD package [3, 16, 116], version 3.6.1 (C++ version linked to Matlab via a mex interface), where we enabled the

option `DISABLE MODELS`, meaning that no modeling is used in MADS, both in the search step and in the construction or order of usage of directions in the poll step.The models are disabled since our solvers at this stage are not using any modeling to speed up the convergence. The reader is referred to Chapter 6 for model incorporation into direct-search methods.

4.2.1 Algorithmic choices

A number of choices regarding parameters and updates of Algorithm 4.1 were made before the tests were launched.

Regarding initializations, the values of λ and μ and of the initial weights followed the choices in CMA-ES (see [78]):

$$\begin{aligned}
\lambda &= 4 + \text{floor}(3\log(n)),\\
\mu &= \text{floor}(\lambda/2),\\
\omega_0^i &= a_i/(a_1 + \cdots + a_\mu), \text{ with } a_i = \log(\lambda/2 + 1/2) - \log(i), \quad i = 1, \ldots, \mu,
\end{aligned}$$

where floor$(\cdot)$ rounds to the nearest integer. The values for c_1, c_μ, c_C, c_σ, and d_σ are chosen also as in the CMA-ES implementation (see [78]) as

$$\begin{aligned}
c_1 &= 2/((n+1.3)^2 + \mu_f),\\
c_\mu &= \min\{1 - c_1, 2(\mu_f - 2 + 1/\mu_f)/((n+2)^2 + \mu_f)\},\\
c_C &= (4 + \mu_f/n)/(n + 4 + 2\mu_f/n),\\
c_\sigma &= (\mu_f + 2)/(n + \mu_f + 5),\\
d_\sigma &= 1 + 2\max\{0, [(\mu_f - 1)/(n+1)]^{\frac{1}{2}} - 1\} + c_\sigma, \text{ with}\\
\mu_f &= (\omega_0^1 + \cdots + \omega_0^\mu)^2/((\omega_0^1)^2 + \cdots + (\omega_0^\mu)^2).
\end{aligned}$$

The initial step length parameters were set to $\sigma_0 = \sigma_0^{\text{CMA-ES}} = 1$. The forcing function selected was $\rho(\sigma) = 10^{-4}\sigma^2$.

To reduce the step length in unsuccessful iterations we used $\sigma_{k+1} = 0.5\sigma_k$ which corresponds to setting $\beta_1 = \beta_2 = 0.5$. In successful iterations, we used $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\text{CMA-ES}}\}$, in attempt to reset the step length to the ES one whenever possible.

The directions d_k^i, $i = 1, \ldots, \lambda$, were drawn from the multivariate normal distribution $\mathcal{C}_k$ updated by CMA-ES, scaled if necessary to obey the safeguards $d_{\min} \leq \|d_k^i\| \leq d_{\max}$, with $d_{\min} = 10^{-10}$, $d_{\max} = 10^{10}$. In the experiments reported, we have never seen a run where there was a need to impose these safeguards.

Updating the weights in Step 2 of Algorithm 4.1 to enforce (4.1) was not activated. On the one hand, we wanted the least amount of changes in CMA-ES. On the other hand, such an update of the weights in Step 2 did not seem to have a real impact on the results for versions max/max and mean/max, perhaps due to the convexity near the solutions present in many of the problems.

4.2.2 Test problems

Our test set $\mathcal{P}$ is the one suggested in [125] and comprises 22 nonlinear vector functions from the CUTEr collection. The problems in $\mathcal{P}$ are then defined by a vector (k_p, n_p, m_p, s_p) of integers. The integer k_p is a reference number for the underlying CUTEr [71] vector function, n_p is the number of variables, m_p is the number of components $F_1, \ldots, F_{m_p}$ of the corresponding vector function F. The objective function value is then computed as the l_2-norm of the vector function F.

The integer $s_p \in \{0, 1\}$ defines the starting point via $x_0 = 10^{s_p} x_s$, where x_s is the standard CUTEr starting point for the corresponding function. According to [125], the use of $s_p = 1$ is helpful for testing solvers from a more remote starting point since the standard starting point tends to be too close to a solution for many of the problems.

The test set $\mathcal{P}$ is then formed by 53 different problems. No problem is overrepresented in $\mathcal{P}$ in the sense that no function k_p appears more than six times. Moreover, no pair (k_p, n_p) appears more than twice. In all cases,

$$2 \leq n_p \leq 12, \quad 2 \leq m_p \leq 65, \quad p = 1, \ldots, 53,$$

with $n_p \leq m_p$. Table 4.1 contains the distribution of n_p across the problems. For other details see [125].

n_p	2	3	4	5	6	7	8	9	10	11	12
Number of problems	5	6	5	4	4	5	6	5	4	4	5

TABLE 4.1: The distribution of n_p in the test set.

The test problems have been considered in four different types, each having 53 instances: smooth (least squares problems obtained from applying the ℓ_2 norm to the vector functions); nonstochastic noisy (obtained by adding oscillatory noise to the smooth ones); piecewise smooth (as in the smooth case but using the ℓ_1 norm instead); stochastic noisy (obtained by adding random noise to the smooth ones).

4.2.3 Test strategies

For our numerical experiments, we chose to work with two types of profiles, data and performance profiles.

Data profiles

Data profiles [125] were designed for derivative-free optimization and show how well a solver performs, given some computational budget, when asked to reach a specific reduction in the objective function value, measured by

$$f(x_0) - f(x) \;\geq\; (1-\alpha)[f(x_0) - f_L],$$

where $\alpha \in (0,1)$ is the level of accuracy, x_0 is the initial iterate, and f_L is the best objective value found by all solvers tested for a specific problem within a given maximal computational budget. In derivative-free optimization, such budgets are typically measured in terms of the number of objective function evaluations.

Data profiles plot the percentage of problems solved by the solvers under consideration for different values of the computational budget. These budgets are expressed in number of points $(n+1)$ required to form a simplex set, allowing the combination of problems of different dimensions in the same profile. Note that a different function of n could be chosen, but $n+1$ is natural in derivative-free optimization (since it is the minimum number of points required to form a positive basis, a simplex gradient, or a model with first-order accuracy).

We used in our experiments a maximal computational budget consisting of $50n$ function evaluations, as we are primarily interested in the behavior of the algorithms for problems where the evaluation of the objective function is expensive. As for the levels of accuracy, we chose two values, $\alpha = 10^{-3}$ and $\alpha = 10^{-7}$. Since the best objective value f_L is chosen as the best value found by all solvers considered, but under a relatively low maximal computational budget, it makes some sense to consider a high accuracy level (like 10^{-7} or less).

Performance profiles

Performance profiles [60] are defined in terms of a performance measure $t_{p,s} > 0$ obtained for each problem $p \in \mathcal{P}$ and solver $s \in \mathcal{S}$. For example, this measure could be based on the amount of computing time or the number of function evaluations required to satisfy

a convergence test. Larger values of $t_{p,s}$ indicate worse performance. For any pair (p, s) of problem p and solver s, the performance ratio is defined by

$$r_{p,s} \;=\; \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}}.$$

The performance profile of a solver $s \in \mathcal{S}$ is then defined as the fraction of problems where the performance ratio is at most τ, that is,

$$\rho_s(\tau) \;=\; \frac{1}{|\mathcal{P}|}\text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\},$$

where $|\mathcal{P}|$ denotes the cardinality of $\mathcal{P}$. Performance profiles seek to capture how well the solver $s \in \mathcal{S}$ performs relatively to the others in $\mathcal{S}$ for all the problems in $\mathcal{P}$. Note, in particular, that $\rho_s(1)$ is the fraction of problems for which solver $s \in \mathcal{S}$ performs the best (efficiency), and that for τ sufficiently large, $\rho_s(\tau)$ is the fraction of problems solved by $s \in \mathcal{S}$ (robustness). In general, $\rho_s(\tau)$ is the fraction of problems with a performance ratio $r_{p,s}$ bounded by τ, and thus solvers with higher values for $\rho_s(\tau)$ are preferable. In this thesis, the performance profiles are plotted in a $\log_2$-scale to better visualize the relative efficiencies of the solvers ($\tau = 1$ will then correspond to $\tau = 0$).

It was suggested in [61] to use the same (scale invariant) convergence test for all solvers compared using performance profiles. The convergence test used in our experiments was

$$f(x) - f_* \;\leq\; \alpha(|f_*| + 1), \tag{4.7}$$

where α is an accuracy level and f_* is an approximation for the optimal value of the problem being tested. The convention $r_{p,s} = +\infty$ is used when the solver s fails to satisfy the convergence test on problem p. We computed f_* as the best objective function value found by the four CMA-ES solvers (our three modified versions and the pure one) using an extremely large computational budget (a number of function evaluations equal to 500000). Thus, in this case, and as opposed to the data profiles case, it makes more sense not to select the accuracy level too small, and our tests were performed with $\alpha = 10^{-2}, 10^{-4}$. The performance profiles were then computed for a maximum of 1500 function evaluations.

4.2.4 Numerical results

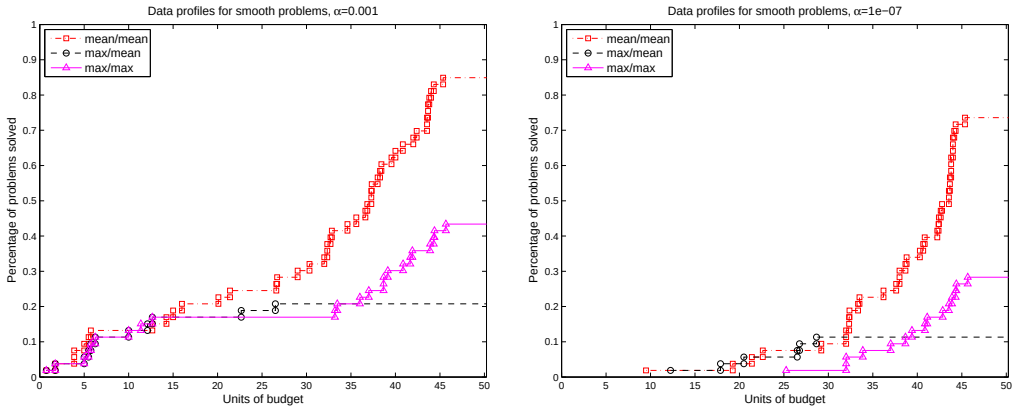
Comparison of the three modified versions of CMA-ES

The purpose of this section is to compare the three modified versions of CMA-ES (mean/mean, max/max, and max/mean) to each other. Our experiments have shown

that the mean/mean version emerges as the best one.

We report here only the results for the class of smooth problems of Section 4.2.2, since the results of the other class problems followed a very similar trend (See Appendix A).

Figure 4.2 depicts the data profile using two levels of accuracy 10^{-3} and 10^{-7}. The data profiles are clearly favorable to the mean/mean version. For instance, with an accuracy of 10^{-3} and within a unit budget of 40, i.e., 40(n+1) function evaluations, the mean/mean version is able to solve about 70% of the problems when the max/max version is solving around 35%. The max/mean version shows the worst profile by solving no more than 20%. The advantage of the mean/mean version for higher accuracy, i.e., 10^{-7}, is more obvious.



(a) Accuracy level of 10^{-3}.

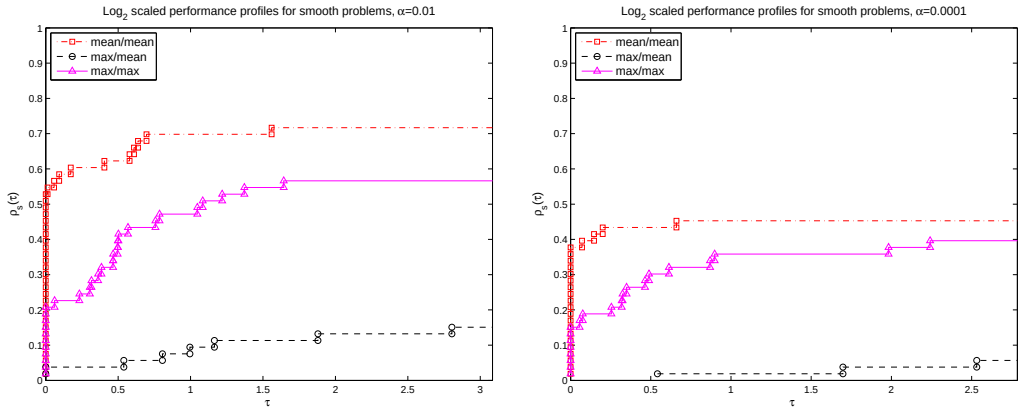
(b) Accuracy level of 10^{-7}.

FIGURE 4.2: Data profiles computed for the set of smooth problems, considering the two levels of accuracy, 10^{-3} and 10^{-7} (for the three modified versions).

Figure 4.3 depicts the performance profiles for smooth problems using two levels of accuracy 10^{-2} and 10^{-4}. Again, the mean/mean version emerges as the best one and outperforms all the other versions in efficiency as well as in robustness. For instance, with an accuracy 10^{-2} of the mean/mean version is able to solve, i.e, $\log_2(\tau) = 0$, more than 50% of the problems, the max/max version is solving around 20% of the problems. The max/mean version is showing the worst profile by solving less then 5% of the problems. The Robust behavior of the mean/mean version is clear as far as the value of τ is getting higher, by attaining a robustness of about 75%.

Comparison with other solvers

The previous section showed that the mean/mean version is performing the best among the three versions tested. Thus, in this section only the mean/mean version is used for



(a) Accuracy level of 10^{-2}.

(b) Accuracy level of 10^{-4}.

FIGURE 4.3: Performance profiles computed for the set of smooth problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4} (for the three modified versions).

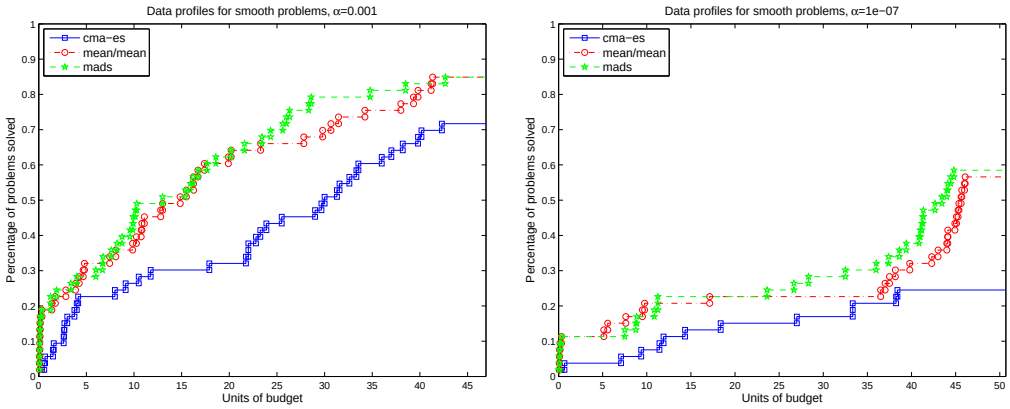
the comparison with the pure CMA-ES and MADS solvers.

Figures 4.4–4.7 report the data profiles obtained by the mean/mean and pure versions and by MADS, for the four types of problems, considering the two different levels of accuracy, $\alpha = 10^{-3}$ and $\alpha = 10^{-7}$ (Figure 4.4: smooth problems; Figure 4.5: nonstochastic noisy problems; Figure 4.6: piecewise smooth problems; Figure 4.7: stochastic noisy problems).

MADS exhibits a slightly better performance than the mean/mean version in the data profiles (which test smaller budgets, i.e., up to 500 function evaluation). But compared to the pure CMA-ES, the mean/mean version is performing significantly better. For instance, by looking to the smooth problems (see Figure 4.4), for an accuracy level of 10^{-3} and for a unit budget of 25, CMA-ES is solving only 45% of the problems when the mean/mean version and MADS are solving around 70% of the tested problems. For high accuracy, CMA-ES shows a significant deterioration compared to the other solvers.

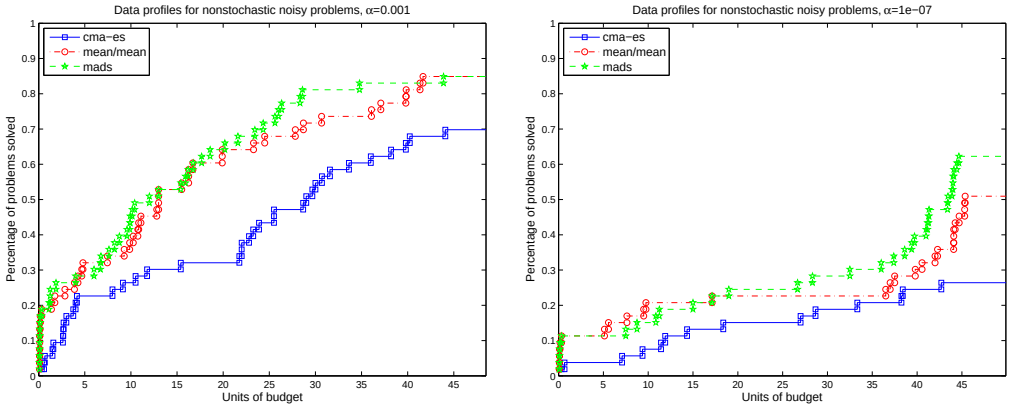
Figures 4.8–4.11 report performance profiles obtained by the mean/mean and pure versions and by MADS, for the four types of problems, considering the two different levels of accuracy, $\alpha = 10^{-2}$ and $\alpha = 10^{-4}$ (Figure 4.8: smooth problems; Figure 4.9: nonstochastic noisy problems; Figure 4.10: piecewise smooth problems; Figure 4.11: stochastic noisy problems).

In terms of performance profiles, the mean/mean version performs roughly the same as MADS in efficiency but better in robustness. For example in Figure 4.8 and for an accuracy of 10^{-2}, the mean/mean and MADS solvers are able to solve around 40% of



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 4.4: Data profiles computed for the set of smooth problems, considering the two levels of accuracy, 10^{-3} and 10^{-7}.



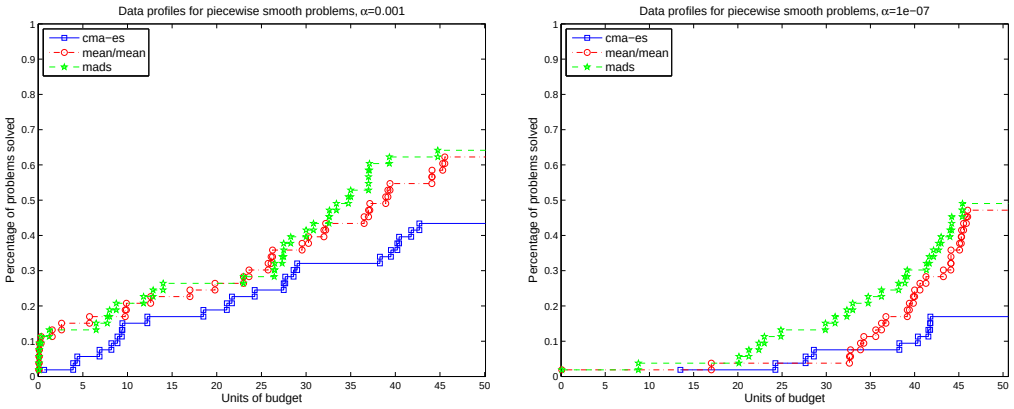
(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 4.5: Data profiles computed for the set of nonstochastic noisy problems, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

smooth problems when CMA-ES is solving 10% of the same test problems. In terms of Robustness, MADS show the worst performance over the other solvers. The advantage of the mean/mean version over the pure CMA-ES is obvious, with the exception of the piecewise problems where the pure CMA-ES overcomes in terms of robustness the mean/mean version (see Figure 4.10).

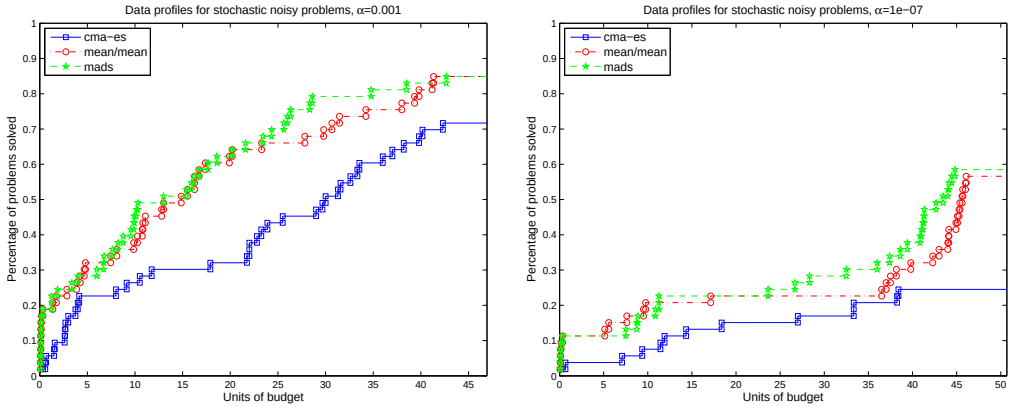
4.2.5 Global optimization tests

In this section we are interested in assessing the impact of our modifications on the ability of CMA-ES to identify the global minimum on problems with a high number of different local minimizers.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 4.6: Data profiles computed for the set of piecewise smooth problems, considering the two levels of accuracy, 10^{-3} and 10^{-7}.



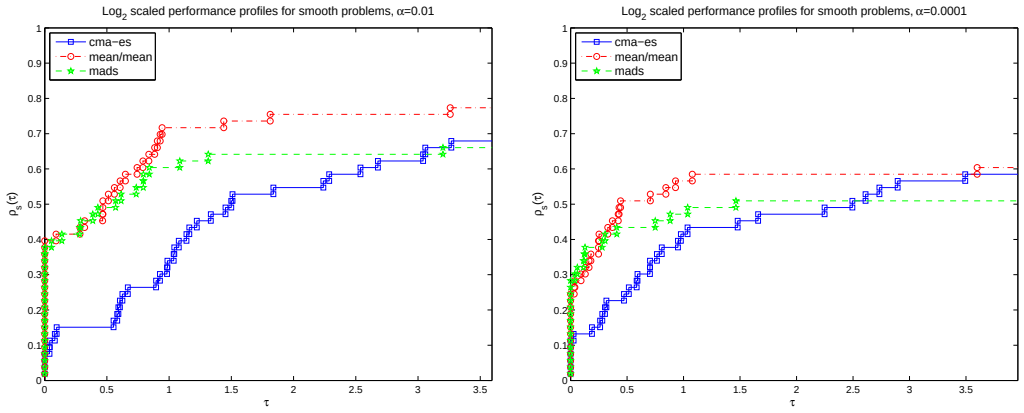
(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 4.7: Data profiles computed for the set of stochastic noisy problems, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

We recall that the mean/mean version exhibited the best performance among the three modified versions of CMA-ES on the test set mentioned in Section 4.2.2. Therefore in this section we will report a comparison of CMA-ES only against this version.

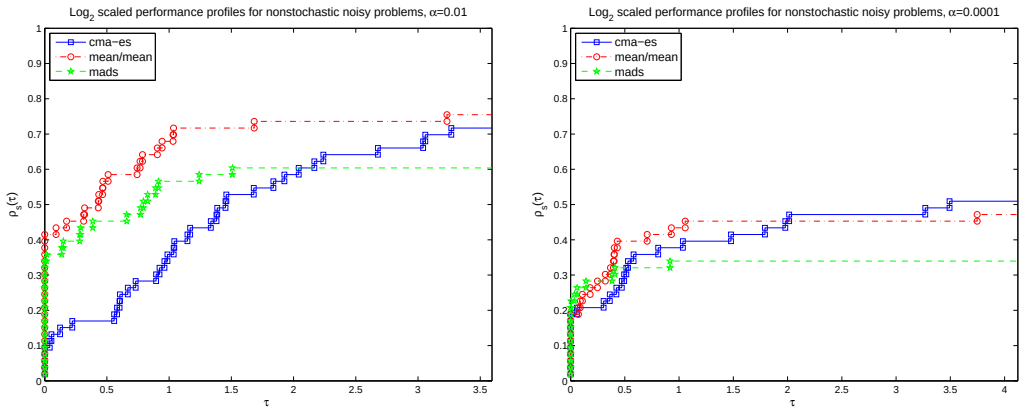
The test set is now composed of the 19 highly multi-modal problems used in [80, 81], where the last 9 are noisy (see Tables 4.2–4.3). We selected dimensions $n = 10, 20$, and, for each dimension, population sizes of $\lambda = 2n, 10n$. For each case and using a large maximal computational budget, we ran our mean/mean CMA-ES version and pure CMA-ES, from 20 different starting points randomly chosen using the Matlab function `randn`. We then computed the median of all the 20 'optimal' values found for each algorithm as well as the median of the respective number of function evaluations taken.

Each run was ended when the function value falls below a certain fitness value, chosen



(a) Accuracy level of 10^{-2}. (b) Accuracy level of 10^{-4}.

FIGURE 4.8: Performance profiles computed for the set of smooth problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4}.



(a) Accuracy level of 10^{-2}. (b) Accuracy level of 10^{-4}.

FIGURE 4.9: Performance profiles computed for the set of nonstochastic noisy problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4}.

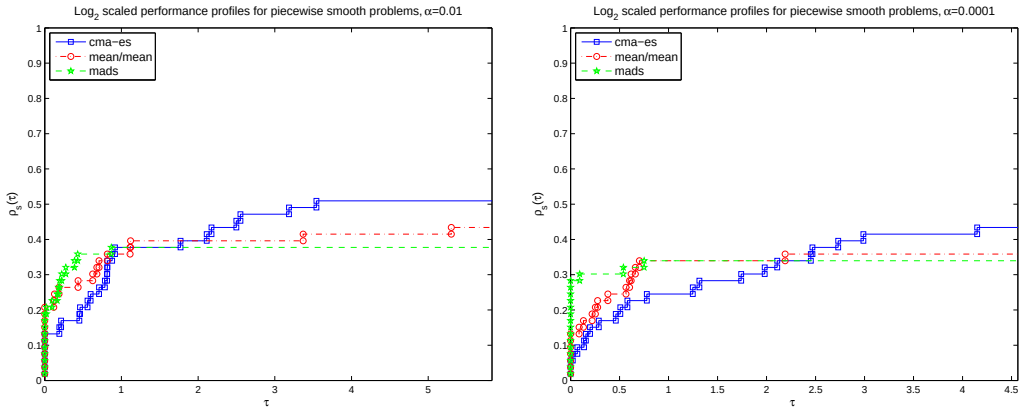
Problem Number	1	2	3	4	5	6	7	8	9	10
Problem index in [81]	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}	f_{23}	f_{24}

TABLE 4.2: Noiseless problems.

Problem Number	11	12	13	14	15	16	17	18	19
Problem index in [80]	f_{122}	f_{123}	f_{124}	f_{125}	f_{126}	f_{127}	f_{128}	f_{129}	f_{130}

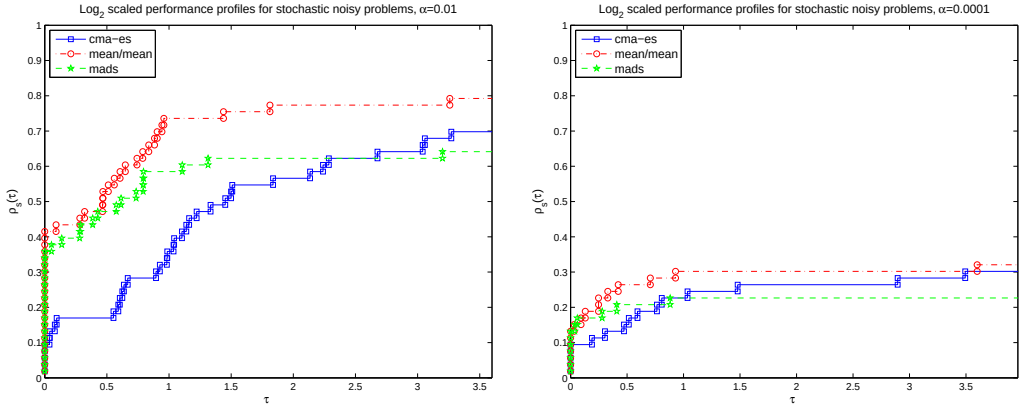
TABLE 4.3: Noisy problems.

as $f_* + 10^{-7}$, where f_* is the optimal value of the corresponding problem, or when the number of function evaluations reaches 250000. To avoid division by large numbers we also stop a run once σ_k becomes smaller than 10^{-10}. It must be made clear that this last criterion makes our versions (in particular the mean/mean one) more parsimonious in terms of function evaluations but it may also possibly restrict the search of the global



(a) Accuracy level of 10^{-2}. (b) Accuracy level of 10^{-4}.

FIGURE 4.10: Performance profiles computed for the set of piecewise smooth problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4}.

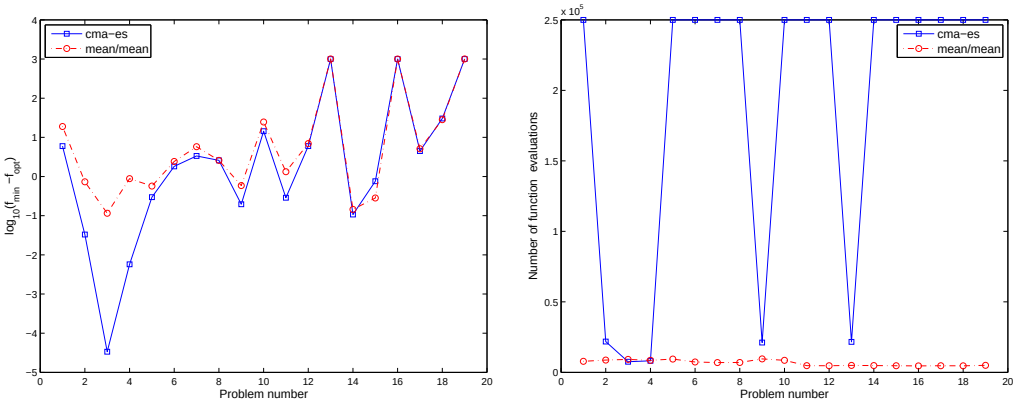


(a) Accuracy level of 10^{-2}. (b) Accuracy level of 10^{-4}.

FIGURE 4.11: Performance profiles computed for the set of stochastic noisy problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4}.

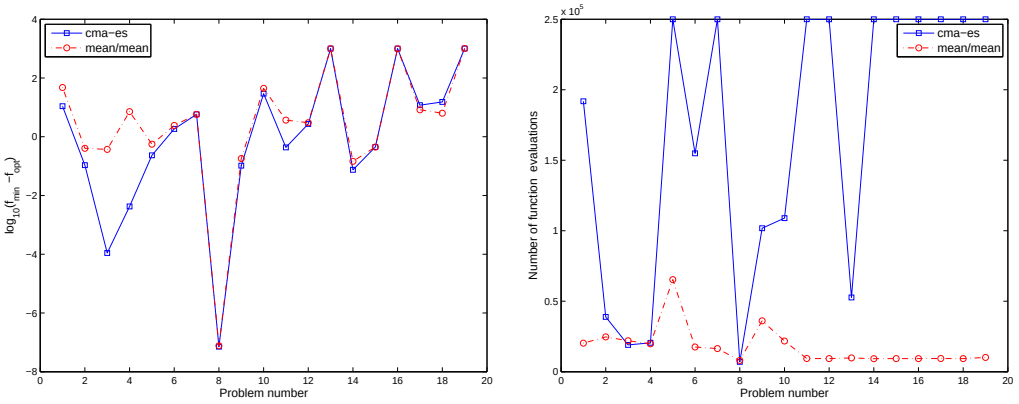
minimum. Note also that the budget is therefore large and the tolerances small since we are interested in observing the asymptotic ability to determine a global minimum (such choices are not likely to be affordable in practical application problems where the objective function is expensive to evaluate).

Figures 4.12(a), 4.13(a), 4.14(a), and 4.15(a) show the median best objective value obtained by the mean/mean and the pure CMA-ES versions, as well as the global optimal value, for all problem dimensions and population sizes and using a $\log_{10}$-scale. Figures 4.12(b), 4.13(b), 4.14(b), and 4.15(b) plot the corresponding median number of objective function evaluations taken. One can see that the pure version of CMA-ES behaves slightly better, when accurately searching for a global minimizer, in particular if a larger population size is given. The two approaches, however, exhibit difficulties in



(a) Best function values (median). (b) Number of function evaluations taken (median).

FIGURE 4.12: Results for the mean/mean version, CMA-ES, and MADS on a set of multi-modal functions of dimension 10 (using $\lambda = 20$).

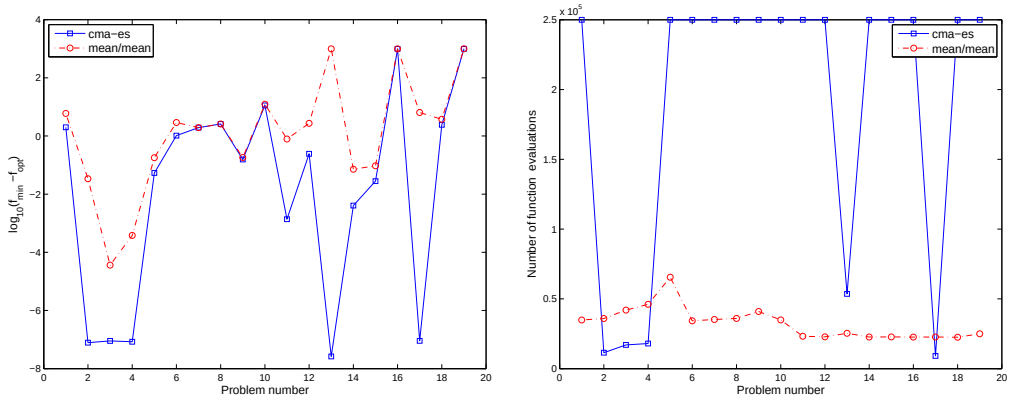


(a) Best function values (median). (b) Number of function evaluations taken (median).

FIGURE 4.13: Results for the mean/mean version, CMA-ES, and MADS on a set of multi-modal functions of dimension 20 (using $\lambda = 40$).

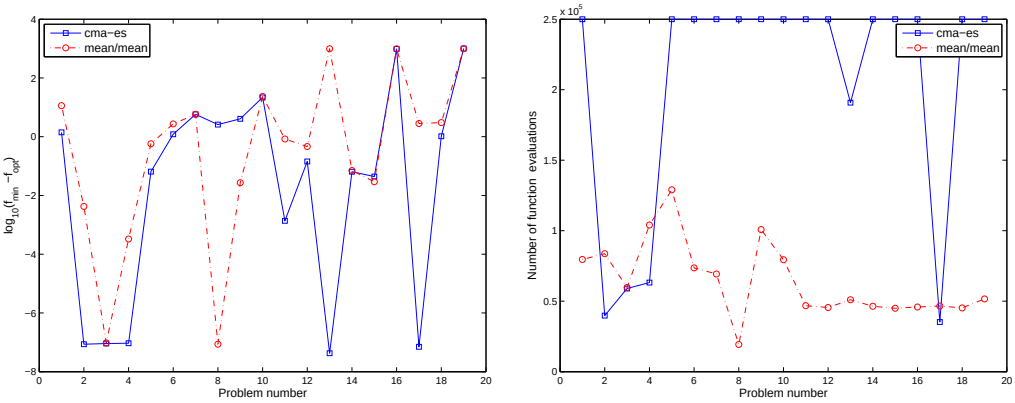
identifying a global minimizer in most of the problems within the given budget. The difficulty of this test set in terms of global optimization calls perhaps for additional algorithmic features such as a multistart technique.

The results showed that CMA-ES cannot handle successfully all problems with many local minimizers and that our modifications (based on classical and rigorous non linear programming globalization techniques) do not change much that state of affairs. A new variant of CMA-ES has been proposed called IPOP-CMA-ES [22] to handle multimodal test functions. IPOP-CMA-ES is based on a restart strategy of CMA-ES with increasing the population size. The advantage of the new variant IPOP-CMA-ES is only effective for relatively small dimensions (up to 50 variables) [22].



(a) Best function values (median). (b) Number of function evaluations taken (median).

FIGURE 4.14: Results for the mean/mean version, CMA-ES, and MADS on a set of multi-modal functions of dimension 10 (using $\lambda = 100$).



(a) Best function values (median). (b) Number of function evaluations taken (median).

FIGURE 4.15: Results for the mean/mean version, CMA-ES, and MADS on a set of multi-modal functions of dimension 20 (using $\lambda = 200$).

4.3 Conclusions

The main contribution of this chapter is to show a possible way to modify a type of ES algorithms, so that they converge to stationary points without any assumption on the starting point. The modified versions of ES promote smaller steps when the larger steps are uphill and thus lead to an improvement in the efficiency of the algorithms in the search of a stationary point. The so-called mean/mean version, where the step is reduced whenever the objective value of the weighted mean of the best trial offspring does not sufficiently reduce the objective value at the current weighted mean, has emerged as the best modified version in our numerical experiments. Apparently, the promotion of such smaller steps has not changed too much the search for the global minimizer in problems with several local minimizers (see Section 4.2.5).

Our approach applies to all ES of the type $(\mu/\mu_W, \lambda)$–ES, although we only used CMA-ES in our numerical tests. A number of issues regarding the interplay of our ES modifications (essentially the step size update based on different sufficient decrease conditions) and the CMA scheme to update the covariance matrix and corresponding step size must be better understood and investigated. In addition, we have not explored to our benefit any hidden ability of the CMA scheme to approximate or predict first or second order information (which might be used in the sufficient decrease conditions or to guide the offspring generation).

It is possible to significantly improve the numerical performance of ES's by incorporating a search step at the beginning of each iteration (as in the search-poll framework of direct search [36]). In such a step, one can, for instance, build a quadratic model using all or some of the points where the objective function has been previously evaluated and then minimize such a model in a certain region (see [53]). The application of such search steps to ES's as well as the extension to the constrained setting will be addressed in the forthcoming chapters.

Chapter 5

Extension to Constraints

In this chapter, we propose a new approach to extend ES to handle general constrained optimization. Under appropriate assumptions, the proposed ES is globally convergent regardless of the starting points. In the general context of ES, various algorithms have been proposed to handle constraints. Coello [44] and Kramer [112] provide a comprehensive survey of the most popular constrained optimization methods currently used within ES. Most approaches use penalty functions [144], where a term penalizing infeasibility is added to the objective function. Other more sophisticated approaches are based on the use of multiobjective optimization [66] or biologically inspired techniques [67, 143]. In this chapter, recall the regarded optimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega = \Omega_r \cap \Omega_{nr}. \end{aligned} \tag{5.1}$$

The feasible region $\Omega \in \mathbb{R}^n$ of this problem is defined by relaxable and/or non-relaxable constraints. In our notation Ω_r is the set of relaxable constraints, which is assumed to be of the form:

$$\Omega_r = \{x \in \mathbb{R}^n : c_i(x) \leq 0, \forall i \in \mathcal{I}\}.$$

No violation is allowed when the constraints are non-relaxable Ω_{nr}, it needs to be satisfied for all the algorithm iterations. Typically, these constraints can be seen as bound or linear constraints. However, the relaxable constraints allowed their violation, and generally need only to be satisfied approximately or asymptotically. The objective function as well as the relaxable constraint function c_i will be assumed to be locally Lipschitz continuous, meaning that both functions are Lipschitz continuous near an accumulation point produced by the algorithm. The proposed algorithm is mainly inspired by recent

works to equip direct search methods with a general procedure to handle both relaxable and unrelaxable constraints [19, 74]. Audet and Dennis [19] outlines a globally convergent general approach based on a progressive barrier, it combines an extreme barrier approach for unrelaxable constraints with non-dominance filters [63]. Vicente and Gratton [74] proposed a second alternative where one handles relaxable constraints by means of a merit function. The latter approach ensures convergence by imposing a sufficient decrease on a merit function (combines information from both objective function and the constraints violation).

For non-relaxable constraints, we address the case where Ω_{nr} is defined by a finite number of linear inequalities, but we will make it precise only later when needed since our theory applies to nonlinear constraints as well. For that purpose, two different feasible approaches are considered. A first one relies on techniques used in directional direct-search methods [52, 108], where one uses an extreme barrier function to prevent infeasible displacements together with the possible use of directions that conform to the local geometry of the feasible region. The extreme barrier function is of the form

$$f_{\Omega_{nr}}(x) \;=\; \begin{cases} f(x) & \text{if } x \in \Omega_{nr}, \\ +\infty & \text{otherwise.} \end{cases} \tag{5.2}$$

($f_{\Omega_{nr}}(x)$ is known as the death penalty function in the terminology of evolutionary algorithms.) We consider that ties of $+\infty$ are broken arbitrarily in the ordering of the offspring samples. The second approach is based first on enforcing all the generated sample points to be feasible, by using a projection mapping of the form:

$$\Phi_{\Omega_{nr}} \;:\; \mathbb{R}^n \;\to\; \Omega_{nr}, \quad \Phi^2_{\Omega_{nr}} \;=\; \Phi_{\Omega_{nr}}. \tag{5.3}$$

The projection is not necessarily the Euclidean one or defined using some other distance, although in the case of bound constraints we will use the ℓ_2-projection (as it is trivial to evaluate) and in the case of general linear constraints we will use the ℓ_1-projection (as it reduces to the solution of an LP). Projection onto the feasible set is regarded as the only alternative to extreme barrier. For general unrelaxable constraints, the projection approach is known to be unpractical and expensive [120].

For relaxable constraints, an augmented Lagrangian approach can be used where one adds a penalty term to the objective function [109, 118]. On the same direction, we propose to adapt the merit function approach for direct-search methods [74] to the evolutionary strategies setting.

For such purpose, we consider the following constraint violation function

$$g(x) = \sum_{i \in \mathcal{I}} \max(c_i(x), 0) \tag{5.4}$$

and the merit function

$$M(x; \delta) = f(x) + \delta \cdot g(x), \tag{5.5}$$

where δ is a positive penalty parameter. The new ES will rely on the merit function to decide and control the distribution of the points. The selection will be based on the value of the merit function at the given points. For relaxable constraints, our convergence theory is globally inspired by [74]. However, the performance of the merit function approach [74] compared to other existing direct search algorithms (e.g. the progressive barrier approach [19]) was not investigated. Thus the contribution of our work compared to [74] is the following: (a) we propose an adaptation of the merit function approach algorithm and convergence theory to the ES setting, (b) we provide a detailed practical implementation for both relaxable and non-relaxable constraints (special care is given to bound and linear non-relaxable constraints), (c) our algorithm is compared to the state of the art DFO algorithms (including global optimization solvers).

The chapter is organized as follows. We start by describing the algorithm as well as the convergence theory behind in Section 5.1. Practical implementation choices are emphasized in Section 5.2. Our numerical experiments comparing the proposed algorithm to other approaches are described in Section 5.3. Finally, in Section 5.4, we draw some conclusions and perspectives.

5.1 A globally convergent ES for general constraints

5.1.1 Algorithm description

The main contribution of Chapter 4 was essentially the monitoring of the quality of the sampling procedure by checking if the objective function has been sufficiently decreased. When that is not the case the step size σ_k is reduced and the iteration becomes unsuccessful. Otherwise, the iteration is successful and the step size σ_k might recover the original ES value σ_k^{ES} if this latter one is sufficiently large. There were different ways to impose sufficient decrease conditions in ES. We will adopt here the version that consists of applying sufficient decrease directly to the weighted mean x_{k+1}^{trial} of the new parents (see Algorithm 4.1), which has been shown to yield global convergence for unconstrained

optimization without any convexity like assumption and to numerically perform the best among the different versions tested.

The extension of the globally convergent ES to the constrained setting follows a hybridization of a feasible approach, where one uses the extreme barrier or the projection approach for relaxable constraints, and of a merit function approach. The trial mean parent x_{k+1}^{trial} will be computed as the weighted mean of the μ best point not in terms of the objective function but regarded to the merit function values of the offspring population. An iteration of the algorithm is considered successful in two situations. The first one is where one has a sufficient decrease in the constraints violation function g, i.e. $g(x_{k+1}^{trial}) < g(x_k) - \rho(\sigma_k)$, and one is sufficiently away from the feasible region, i.e. $g(x_{k+1}^{trial}) > C\rho(\sigma_k)$ for some constant $C > 1$.

The second successful situation is when the merit function is sufficiently decreased, i.e. $M(x_{k+1}^{trial}, \delta_k) < M(x_k, \delta_k) - \rho(\sigma_k)$ for a given choice of the penalty parameter δ_k. The update of δ_k is done on the same manner as [74]

$$\delta_k = \max\left\{\bar{\delta}, \frac{f(x_{k+1}^{trial}) - f(x_k)}{C\rho(\sigma_k)}\right\},$$

where $\bar{\delta} > 0$. Following the same notations used in [74], the trial mean parent will be declared as a successful point if the following procedure is fulfilled:

Begin (successful point).

Given a parent x_k and a step size σ_k, the trial parent x_{k+1}^{trial} is successful if

$$g(x_{k+1}^{trial}) \ < \ g(x_k) - \rho(\sigma_k) \quad \text{and} \quad g(x_k) > C\rho(\sigma_k)$$

or, if that is false, if

$$M(x_{k+1}^{trial}, \delta_k) \ < \ M(x_k, \delta_k) - \rho(\sigma_k),$$

where

$$\delta_k = \max\left\{\bar{\delta}, \frac{f(x_{k+1}^{trial}) - f(x_k)}{C\rho(\sigma_k)}\right\} \tag{5.6}$$

where $\bar{\delta} > 0$ is a sufficiently large penalty parameter.

End (successful point).

Before checking wether the trial point is successful or not, the algorithm will try first to restore the feasibility or at least decrease the constraints violation. A restoration process will be activated when the constraints violation is sufficiently decreasing at a trial point sufficiently away from the feasible region and for which the the objective function has increased. In other words, the restoration is entered if one has $g(x_{k+1}^{trial}) \ < \ g(x_k) - \rho(\sigma_k)$,

$g(x_k) > C\rho(\sigma_k)$, and $M(x_{k+1}^{trial}, \bar{\delta}) \geq M(x_k, \bar{\delta})$.

Begin (Restoration identifier).

Given a parent x_k and a step size σ_k, the trial parent x_{k+1}^{trial} is a Restoration identifier if

$$g(x_{k+1}^{trial}) < g(x_k) - \rho(\sigma_k) \quad \text{and} \quad g(x_k) > C\rho(\sigma_k)$$

and

$$M(x_{k+1}^{trial}, \bar{\delta}) \geq M(x_k, \bar{\delta}).$$

End (Restoration identifier).

Our globally convergent ES is described in details below, in Algorithm 5.1. Note that directions used to compute the offspring are not necessarily the ES directions randomly generated, in what can be seen as a modification made in preparation to what comes next regarding the non-relaxable constraints. We will denote the directions used to compute the offspring by $\tilde{d}_k^i$ (see Section 5.3.1).

The restoration Algorithm tries to reduce the constraints violation. The selection process is then based on the constraints violation function but not on the merit one as in Algorithm 5.1. Restoration is left as far as one is not able anymore to reduce the constraints violation and such as considerable increase in the objective function f is no longer observed. The complete restoration procedure is outlined by Algorithm 5.2.

Under appropriate assumptions we will now prove global convergence of the proposed extension to constraints for ES. Our convergence analysis is inspired by direct-search methods for nonsmooth functions [74].

5.1.2 Step size behavior

Theorem 5.1. *Consider a sequence of iterations generated by the Algorithm 5.1 without any stopping criterion. Let f be bounded below and assuming that Restoration is not entered after a certain order.*

Then,

$$\liminf_{k \to +\infty} \sigma_k = 0.$$

Proof. Suppose that there exists a $\bar{k} > 0$ and $\sigma > 0$ such that $\sigma_k > \sigma$ and k is a Main iteration $k \geq \bar{k}$. If there is an infinite sequence J_1 of successful iterations after $\bar{k}$, this leads to a contradiction to the fact that g and f are bounded below.

In fact, since ρ is a nondecreasing, positive function, $\rho(\sigma_k) \geq \rho(\sigma) > 0$. One has :

Algorithm 5.1: A globally convergent ES for general constraints (Main).

Initialization: Choose positive integers λ and μ such that $\lambda \geq \mu$. Select an initial $x_0 \in \Omega_{nr}$ and evaluate $f(x_0)$. Choose initial step lengths $\sigma_0, \sigma_0^{\mathrm{ES}} > 0$ and initial weights $(\omega_0^1, \ldots, \omega_0^\mu) \in S$. Choose constants $\bar{\delta} > 0$, $C > 1$, $\beta_1, \beta_2, d_{\min}, d_{\max}$ such that $0 < \beta_1 \leq \beta_2 < 1$ and $0 < d_{\min} < d_{\max}$. Select a forcing function $\rho(\cdot)$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. **Offspring Generation:** Compute new sample points $Y_{k+1} = \{y_{k+1}^1, \ldots, y_{k+1}^\lambda\}$ such that
$$y_{k+1}^i \;=\; x_k + \sigma_k \tilde{d}_k^i,\; i = 1, \ldots, \lambda,$$
where the directions $\tilde{d}_k^i$'s are computed from the original ES directions d_k^i's (which in turn are drawn from a chosen ES distribution $\mathcal{C}_k$ and scaled if necessary to satisfy $d_{\min} \leq \|d_k^i\| \leq d_{\max}$).

2. **Parent Selection:** Evaluate $M(y_{k+1}^i, \bar{\delta})$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\lambda\}$ by increasing order: $M(\tilde{y}_{k+1}^1, \bar{\delta}) \leq \cdots \leq M(\tilde{y}_{k+1}^\lambda, \bar{\delta})$.
Select the new parents as the best μ offspring sample points $\{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\mu\}$, and compute their weighted mean
$$x_{k+1}^{trial} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i.$$

3. **Imposing the Merit Function Approach:** If $x_{k+1}^{trial} \notin \Omega_{nr}$ the iteration is declared unsuccessful, otherwise.
If x_{k+1}^{trial} is a **Restoration identifier**, then enter Restoration (with $k_r = k$).
Otherwise, if x_{k+1}^{trial} is a **successful point**, then declare the iteration successful. Otherwise, declare the iteration as unsuccessful.

4. **Updates:** If the iteration is successful then set $x_{k+1} = x_{k+1}^{trial}$, and $\sigma_{k+1} \geq \sigma_k$ (for example $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\mathrm{ES}}\}$). Otherwise set $x_{k+1} = x_k$ and $\sigma_{k+1} = \beta_k \sigma_k$, with $\beta_k \in (\beta_1, \beta_2)$.
Update the ES step length $\sigma_{k+1}^{\mathrm{ES}}$, the distribution $\mathcal{C}_k$, and the weights $(\omega_{k+1}^1, \ldots, \omega_{k+1}^{m_\mu}) \in S$. Increment k and return to Step 1.

If $g(x_{k+1}) \leq g(x_k) - \rho(\sigma_k)$ and $g(x_{k+1}) > C\rho(\sigma_k)$ for all $k \in J_1$, then

$$g(x_{k+1}) \leq g(x_k) - \rho(\sigma),$$

which obviously contradicts the boundedness below of g by 0.

Thus there must exists an infinite subsequence $J_2 \subseteq J_1$ of iterates for which $M(x_{k+1}, \delta_k) < M(x_k, \delta_k) - \rho(\sigma_k)$. Here we consider two cases.

Algorithm 5.2: A globally convergent ES for general constraints (Restoration).

Initialization: Start from $x_{k_r} \in \Omega_{nr}$ given from the Main algorithm and consider the same parameter as in there.

For $k = k_r, k_r + 1, k_r + 2, \ldots$

1. **Offspring Generation:** Compute new sample points $Y_{k+1} = \{y^1_{k+1}, \ldots, y^\lambda_{k+1}\}$ such that
$$y^i_{k+1} \;=\; x_k + \sigma_k \tilde{d}^i_k,\; i = 1, \ldots, \lambda,$$
where the directions $\tilde{d}^i_k$'s are computed from the original ES directions d^i_k's (which in turn are drawn from a chosen ES distribution $\mathcal{C}_k$ and scaled if necessary to satisfy $d_{\min} \le \|d^i_k\| \le d_{\max}$).

2. **Parent Selection:** Evaluate $g(y^i_{k+1})$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}^1_{k+1}, \ldots, \tilde{y}^\lambda_{k+1}\}$ by increasing order: $g(\tilde{y}^1_{k+1}) \le \cdots \le g(\tilde{y}^\lambda_{k+1})$.
Select the new parents as the best μ offspring sample points $\{\tilde{y}^1_{k+1}, \ldots, \tilde{y}^\mu_{k+1}\}$, and compute their weighted mean
$$x^{trial}_{k+1} \;=\; \sum_{i=1}^{\mu} \omega^i_k \tilde{y}^i_{k+1}.$$

3. **Imposing Restoration Condition:** If $x^{trial}_{k+1} \notin \Omega_{nr}$ the iteration is declared unsuccessful, Otherwise.
Declare the iteration successful if one has
$$g(x^{trial}_{k+1}) \;<\; g(x_k) - \rho(\sigma_k) \quad \text{and} \quad g(x_k) > C\rho(\sigma_k)$$
In such a case, set $x_{k+1} = x^{trial}_{k+1}$.
Otherwise, consider the iteration unsuccessful.
Leave Restoration and return to the Main algorithm (starting at a new $(k+1)$-th iteration using x_{k+1} and σ_{k+1}) if the iteration is unsuccessful and $M(x^{trial}_{k+1}, \bar{\delta}) \;<\; M(x_k, \bar{\delta})$

4. **Updates:** As in the Main algorithm.

In the first case, for k sufficiently large, all these iterates are such that $\delta_k = \bar{\delta}$. In such an occurrence one has that

$$M(x_{k+1}, \bar{\delta}) \;<\; M(x_k, \bar{\delta}) - \rho(\sigma_k) \;\le\; M(x_k, \bar{\delta}) - \rho(\sigma)\; \forall k \in J_2.$$

However, in the successful iterations where $g(x_{k+1}) < g(x_k) - \rho(\sigma_k)$ and $g(x_{k+1}) > C\rho(\sigma_k)$, since the restoration was not entered, one knows that $M(x_{k+1}, \bar{\delta}) \;<\; M(x_k, \bar{\delta})$. Thus $M(x_k, \bar{\delta})$ tends to $-\infty$ which is a contradiction, since both f and g are bounded below.

The second case, there is an infinite number of iterations in J_2 such that

$$\delta_k \;=\; \frac{f(x_{k+1}) - f(x_k)}{C\rho(\sigma_k)}.$$

For these iterations, one has either $g(x_{k+1}) \geq g(x_k) - \rho(\sigma_k)$ or $g(x_{k+1}) \leq C\rho(\sigma_k)$. Thus, since $C > 1$, one has either

$$f(x_{k+1}) - f(x_k) \;=\; \delta_k C\rho(\sigma_k) \;\geq\; \delta_k[g(x_k) - g(x_{k+1})]$$

or

$$f(x_{k+1}) - f(x_k) \;=\; \delta_k C\rho(\sigma_k) \;\geq\; \delta_k g(x_k) \;\geq\; \delta_k[g(x_k) - g(x_{k+1})],$$

both leading to $M(x_{k+1}, \delta_k) \;\geq\; M(x_k, \bar{\delta}_k)$ which contradicts $M(x_{k+1}, \delta_k) \;<\; M(x_k, \delta_k) - \rho(\sigma_k)$. The proof is thus completed if there is an infinite number of successful iterations. However, if no more successful iterations occur after a certain order, then this also leads to a contradiction. The conclusion is that one must have a subsequence of iterations driving σ_k to zero. □

Theorem 5.2. *Consider a sequence of iterations generated by Algorithm 5.1 without any stopping criterion. Let f be bounded below and assuming that Restoration is not entered after a certain order.*

There exists a subsequence K of unsuccessful iterates for which $\lim_{k\in K} \sigma_k = 0$ (i.e. refining subsequence).

And if the sequence $\{x_k\}$ is bounded, then there exists an x_* and a refining subsequence K such that $\lim_{k\in K} x_k = x_*$.

Proof. From Theorem 5.1, there must exist an infinite subsequence K of unsuccessful iterates for which σ_{k+1} goes to zero. In a such case we have $\sigma_k = (1/\beta_k)\sigma_{k+1}$, $\beta_k \in (\beta_1, \beta_2)$, and $\beta_1 > 0$, and thus $\sigma_k \to 0$, for $k \in K$, too.

The second part of the Theorem is also easily proved by extracting a convergent subsequence of the subsequence K of the first part for which x_k converges to x_*. □

5.1.3 Global convergence

The global convergence is achieved by establishing that some type of directional derivatives are nonnegative at limit points of refining subsequences along refining directions (see Section 2.11). When h is Lipschitz continuous near $x_* \in \Omega_{nr}$, one can make use of

the Clarke-Jahn generalized derivative along a direction d

$$h^\circ(x_*; d) = \limsup_{\substack{x \to x_*, x \in \Omega_{nr} \\ t \downarrow 0, x + td \in \Omega_{nr}}} \frac{h(x + td) - h(x)}{t}.$$

(Such a derivative is essentially the Clarke generalized directional derivative [43], adapted by Jahn [98] to the presence of constraints). However, for the proper definition of $h^\circ(x_*; d)$, one needs to guarantee that $x + td \in \Omega_{nr}$ for $x \in \Omega_{nr}$ arbitrarily close to x_* which is assured if d is hypertangent to Ω_{nr} at x_*. In the following, $B(x; \Delta)$ is the closed ball formed by all points which dist no more than Δ to x.

Definition 5.3. A vector $d \in \mathbb{R}^n$ is said to be a hypertangent vector to the set $\Omega_{nr} \subseteq \mathbb{R}^n$ at the point x in Ω_{nr} if there exists a scalar $\epsilon > 0$ such that

$$y + tw \in \Omega_{nr}, \quad \forall y \in \Omega_{nr} \cap B(x; \epsilon), \quad w \in B(d; \epsilon), \quad \text{and} \quad 0 < t < \epsilon.$$

The hypertangent cone to Ω at x, denoted by $T^H_{\Omega_{nr}}(x)$, is then the set of all hypertangent vectors to Ω_{nr} at x. Then, the Clarke tangent cone to Ω_{nr} at x (denoted by $T^{Cl}_{\Omega_{nr}}(x)$) can be defined as the closure of the hypertangent cone $T^H_{\Omega_{nr}}(x)$ (when the former is nonempty, an assumption we need to make for global convergence anyway). The Clarke tangent cone generalizes the notion of tangent cone in Nonlinear Programming [130], and the original definition $d \in T^{Cl}_{\Omega_{nr}}(x)$ is given below.

Definition 5.4. A vector $d \in \mathbb{R}^n$ is said to be a Clarke tangent vector to the set $\Omega_{nr} \subseteq \mathbb{R}^n$ at the point x in the closure of Ω_{nr} if for every sequence $\{y_k\}$ of elements of Ω_{nr} that converges to x and for every sequence of positive real numbers $\{t_k\}$ converging to zero, there exists a sequence of vectors $\{w_k\}$ converging to d such that $y_k + t_k w_k \in \Omega_{nr}$.

Given a direction v in the tangent cone, possibly not in the hypertangent one, one can consider the Clarke-Jahn generalized derivative to Ω_{nr} at x_* as the limit

$$h^\circ(x_*; v) = \lim_{d \in T^H_{\Omega_{nr}}(x_*), d \to v} h^\circ(x_*; d)$$

(see [18]). A point $x_* \in \Omega_{nr}$ is considered Clarke stationary if $h^\circ(x_*; d) \geq 0$, $\forall d \in T^{Cl}_{\Omega_{nr}}(x_*)$.

Assuming restoration is never entered after a certain order

Theorem 5.5. *Consider the algorithm 5.1 and let* $a_k = \sum_{i=1}^{\mu} \omega_k^i \tilde{d}_k^i$. *Assume that* f *is bounded below. Assume that Restoration is not entered after a certain order.*

Let $x_* \in \Omega_{nr}$ be the limit point of a convergent subsequence of unsuccessful of iterates $\{x_k\}_K$ for which $\lim_{k\in K} \sigma_k = 0$. Assume that g is Lipschitz continuous near x_* with constant $\nu_g > 0$.

If $d \in T^H_{\Omega_{nr}}(x_*)$ is a refining direction associated with $\{a_k/\|a_k\|\}_K$, then either $g(x_*) = 0$ (implying $x_* \in \Omega_r$ and thus $x_* \in \Omega$) or $g^\circ(x_*; d) \geq 0$.

Proof. Let d be a limit point of $\{a_k/\|a_k\|\}_K$. Then it must exist a subsequence of K' of K such that $a_k/\|a_k\| \to d$ on K'. On the other hand, we have for all k that

$$x_{k+1}^{trial} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i \;=\; x_k + \sigma_k \sum_{i=1}^{\mu} \omega_k^i \tilde{d}_k^i \;=\; x_k + \sigma_k a_k.$$

Since the iteration $k \in K'$ is unsuccessful, $g(x_{k+1}^{trial}) \geq g(x_k) - \rho(\sigma_k)$ or $g(x_{k+1}^{trial}) \leq C\rho(\sigma_k)$, and then either there exists an infinite number of the first or the second. In the later case, there exists a subsequence $K_1 \subseteq K'$ such that $g(x_{k+1}^{trial}) \leq C\rho(\sigma_k)$, it is trivial to obtain $g(x_*) = 0$ using both the continuity of g and the fact that σ_k tends to zero in K_1.

In the former case, there exists a subsequence $K_2 \subseteq K'$ such that the sequence $\{\dfrac{a_k}{\|a_k\|}\}_{k\in K_2}$ converges to $d \in T^H_{\Omega_{nr}}(x_*)$ in K_2 and the sequence $\{\|a_k\|\sigma_k\}_{k\in K_2}$ goes to zero in K_2 (a_k is bounded above for all k, and so $\sigma_k\|a_k\|$ tends to zero when σ_k does). Thus one must have necessarily for k sufficiently large in K_2, $x_k + \sigma_k a_k \in \Omega_{nr}$ such that

$$g(x_k + \sigma_k a_k) \geq g(x_k) - \rho(\sigma_k).$$

Thus, from the definition of the Clarke-Jahn generalized derivative along the directions $d \in T^H_{\Omega_{nr}}(x_*)$,

$$\begin{aligned} g^\circ(x_*; d) &= \limsup_{\substack{x \to x_*, x \in \Omega_{nr} \\ t \downarrow 0, x + td \in \Omega_{nr}}} \frac{g(x + td) - g(x)}{t} \\ &\geq \limsup_{k\in K_2} \frac{g(x_k + \sigma_k\|a_k\|d) - g(x_k)}{\sigma_k\|a_k\|} \\ &\geq \limsup_{k\in K_2} \frac{g(x_k + \sigma_k\|a_k\|(a_k/\|a_k\|)) - g(x_k)}{\sigma_k\|a_k\|} - g_k, \end{aligned}$$

where,

$$g_k \;=\; \frac{g(x_k + \sigma_k a_k) - g(x_k + \sigma_k\|a_k\|d)}{\sigma_k\|a_k\|}.$$

From the Lipschitz continuity of g near x_* one has

$$\begin{aligned} g_k &= \frac{g(x_k + \sigma_k a_k) - g(x_k + \sigma_k \|a_k\| d)}{\sigma_k \|a_k\|} \\ &\leq \nu_g \left\| \frac{a_k}{\|a_k\|} - d \right\| \end{aligned}$$

tends to zero on K_2. Finally,

$$\begin{aligned} g^\circ(x_*; d) &\geq \limsup_{k \in K_2} \frac{g(x_k + \sigma_k a_k) - g(x_k) + \rho(\sigma_k)}{\sigma_k \|a_k\|} - \frac{\rho(\sigma_k)}{\sigma_k \|a_k\|} - g_k \\ &= \limsup_{k \in K_2} \frac{g(x_k + \sigma_k a_k) - g(x_k) + \rho(\sigma_k)}{\sigma_k \|a_k\|}. \end{aligned}$$

One then obtains $g^\circ(x_*; d) \geq 0$. □

When the refining directions are dense in $T^{Cl}_{\Omega_{nr}}(x_*) \cap \{d \in \mathbb{R}^n : \|d\| = 1, \}$, the limit point x_* will be Clarke stationary for the constraint violation problem:

$$\begin{array}{ll} \min & g(x) \\ \text{s.t.} & x \in \Omega_{nr} \end{array}. \tag{5.7}$$

Theorem 5.6. *Consider the algorithm 5.1 and let* $a_k = \sum_{i=1}^{\mu} \omega_k^i \tilde{d}_k^i$. *Assume that* f *is bounded below. Assume that Restoration is not entered after a certain order.*

Let $x_* \in \Omega$ *be the limit point of a convergent subsequence of unsuccessful iterates* $\{x_k\}_{k \in K}$ *for which* $\lim_{k \in K} \sigma_k = 0$. *Assume that* g *and* f *are Lipschitz continuous near* x_*.

Assume that $T^{Cl}_{\Omega_{nr}}(x_*)$ *has a non-empty interior.*

Then either $g(x_*) = 0$ *(implying* $x_* \in \Omega$ *) or if the set of refining directions associated with* $K' \subset K$ *and* x_* *is dense in* $T^{Cl}_{\Omega_{nr}}(x_*) \cap \{d \in \mathbb{R}^n : \|d\| = 1, \}$, *then* $g^\circ(x_*; v) \geq 0$ *for all* $v \in T^{Cl}_{\Omega_{nr}}(x_*)$, *and* x_* *is a stationary point of the constraint violation problem (5.1).*

Proof. As outlined in the proof of Theorem 5.5, if there exists an infinite number of cases such that $g(x_{k+1}^{trial}) \leq C\rho(\sigma_k)$, it is trivial then to obtain $g(x_*) = 0$ using both the continuity of g and the fact that σ_k tends to zero in K.

Let $v \in T^{Cl}_{\Omega_{nr}}(x_*)$ and $\|v\| = 1$, then v is the limit of a sequence $\mathcal{D}$ of refining directions d with K' and x_* such that $d \in T^H_{\Omega_{nr}}$. For each direction d one applies Theorem 5.5 to obtain $g^\circ(x_*; d) \geq 0$. Thus $g^\circ(x_*; v) = \lim_{d \in T^H_{\Omega_{nr}}, d \in \mathcal{D}} g^\circ(x_*; d) \geq 0$. For non-normalized v the result holds since $T^{Cl}_{\Omega_{nr}}(x_*)$ is a cone and the Clarke derivatives are homogeneous in their second arguments. □

For an intermediate optimality result. One can notice that we do not use $x_* \in \Omega_r$ explicitly in the proof, but one notes that $g^\circ(x_*; d) \leq 0$ only describes the cone of first order linearized directions under feasibility assumption $x_* \in \Omega_r$.

Theorem 5.7. *Consider the algorithm 5.1 and let* $a_k = \sum_{i=1}^{m_\mu} \omega_k^i \tilde{d}_k^i$. *Assume that* f *is bounded below. Assume that Restoration is not entered after a certain order.*

Let $x_* \in \Omega_{nr}$ *be the limit point of a convergent subsequence of unsuccessful of iterates* $\{x_k\}_K$ *for which* $\lim_{k\in K} \sigma_k = 0$. *Assume that* g *and* f *are Lipschitz continuous near* x_*.

If $d \in T^H_{\Omega_{nr}}(x_*)$ *is a refining direction associated with* $\{a_k/\|a_k\|\}_K$ *such that* $g^\circ(x_*; d) \leq 0$. *Then* $f^\circ(x_*; d) \geq 0$.

Proof. By assumption there exists a subsequence $K' \subseteq K$ such that the sequence $\{a_k/\|a_k\|\}_{k\in K'}$ converges to $d \in T^H_{\Omega_{nr}}(x_*)$ in K_2 and the sequence $\{\|a_k\|\sigma_k\}_{k\in K'}$ goes to zero in K', Thus one must have necessarily for k sufficiently large in K_2, $x_{k+1}^{trial} = x_k + \sigma_k a_k \in \Omega_{nr}$. Since the iteration $k \in K'$ is unsuccessful, one is sure that δ_k is updated according to (5.6).

If $\delta_k = [f(x_{k+1}^{trial}) - f(x_k)]/[C\rho(\sigma_k)]$, then it is because $[f(x_{k+1}^{trial}) - f(x_k)]/[C\rho(\sigma_k)] \geq \bar{\delta}$, and thus

$$\frac{f(x_k + \sigma_k a_k) - f(x_k)}{\|a_k\|\sigma_k} \geq C\bar{\delta}\frac{\rho(\sigma_k)}{\sigma_k\|a_k\|} \tag{5.8}$$

If not, $\delta_k = \bar{\delta}$, then $M(x_{k+1}^{trial}, \bar{\delta}) \geq M(x_k, \bar{\delta}) - \rho(\sigma_k)$, and thus

$$\frac{f(x_k + \sigma_k a_k) - f(x_k)}{\|a_k\|\sigma_k} \geq -\bar{\delta}\frac{g(x_k + \sigma_k a_k) - g(x_k)}{\|a_k\|\sigma_k} - \frac{\rho(\sigma_k)}{\sigma_k\|a_k\|} \tag{5.9}$$

On the other hand,

$$\begin{aligned} f^\circ(x_*; d) &= \limsup_{\substack{x \to x_*, x \in \Omega_{nr} \\ t \downarrow 0, x + td \in \Omega_{nr}}} \frac{f(x + td) - f(x)}{t} \\ &\geq \limsup_{k\in K'} \frac{f(x_k + \sigma_k\|a_k\|d) - f(x_k)}{\sigma_k\|a_k\|} \\ &\geq \limsup_{k\in K'} \frac{f(x_k + \sigma_k\|a_k\|(a_k/\|a_k\|)) - f(x_k)}{\sigma_k\|a_k\|} - f_k, \end{aligned}$$

where,

$$f_k = \frac{f(x_k + \sigma_k a_k) - f(x_k + \sigma_k \|a_k\| d)}{\sigma_k \|a_k\|},$$

which then implies from (5.9)

$$\begin{aligned} f^\circ(x_*; d) &\geq \limsup_{k \in K'} \frac{f(x_k + \sigma_k \|a_k\|(a_k/\|a_k\|)) - f(x_k)}{\sigma_k \|a_k\|} - f_k, \\ &\geq \limsup_{k \in K'} -\bar{\mu} \frac{g(x_k + \sigma_k a_k) - g(x_k)}{\|a_k\| \sigma_k} - \frac{\rho(\sigma_k)}{\sigma_k \|a_k\|} - f_k \\ &\geq \limsup_{k \in K'} -\bar{\mu} \frac{g(x_k + \sigma_k \|a_k\| d) - g(x_k)}{\sigma_k \|a_k\|} + \bar{\mu} g_k - \frac{\rho(\sigma_k)}{\sigma_k \|a_k\|} - f_k, \end{aligned}$$

where

$$g_k = \frac{g(x_k + \sigma_k a_k) - g(x_k + \sigma_k \|a_k\| d)}{\sigma_k \|a_k\|}.$$

From the assumption $g^\circ(x_*; d) \leq 0$, one has

$$\limsup_{k \in K'} \frac{g(x_k + \sigma_k \|a_k\| d) - g(x_k)}{\sigma_k \|a_k\|} \leq \limsup_{\substack{x \to x_*, x \in \Omega_{nr} \\ t \downarrow 0, x + td \in \Omega_{nr}}} \frac{g(x + td) - g(x)}{t} \leq 0,$$

one obtains then

$$f^\circ(x_*; d) \geq \limsup_{k \in K'} \bar{\mu} g_k - \frac{\rho(\sigma_k)}{\sigma_k \|a_k\|} - f_k. \tag{5.10}$$

The Lipschitz continuity of both g and f near x_* guaranties that the quantities f_k and g_k tend to zero in K'. The proof is completed since the right-hand-sides of (5.10) and (5.8) tend to zero in K'. □

Theorem 5.8. *Consider the algorithm 5.1 and let* $a_k = \sum_{i=1}^{m_\mu} \omega_k^i \tilde{d}_k^i$. *Assume that* f *is bounded below. Assume that Restoration is not entered after a certain order.*

Let $x_* \in \Omega$ *be the limit point of a convergent subsequence of unsuccessful iterates* $\{x_k\}_{k \in K}$ *for which* $\lim_{k \in K} \sigma_k = 0$. *Assume that* g *and* f *are Lipschitz continuous near* x_*.

Assume that the set

$$T(x_*) = T^H_{\Omega_{nr}} \cap \{d \in \mathbb{R}^n : \|d\| = 1, g^\circ(x_*, d) \leq 0\} \tag{5.11}$$

has a non-empty interior.

Let the set of refining directions be dense in $T(x_*)$. *Then* $f^\circ(x_*, v) \geq 0$ *for all* $v \in T^{Cl}_{\Omega_{nr}}(x_*)$ *such that* $g^\circ(x_*, v) \leq 0$, *and* x_* *is a Clarke stationary point of (5.1).*

Proof. Let $v \in T^{Cl}_{\Omega_{nr}}(x_*)$ such that $g^\circ(x_*, v) \leq 0$, and $\|v\| = 1$. Then v is the limit of a sequence $\mathcal{D}$ of refining direction d such that $d \in T^H_{\Omega_{nr}}$ and $g^\circ(x_*, d) \leq 0$. For each such d one can apply Theorem 5.7 and obtain $f^\circ(x_*; d) \geq 0$. Thus, $f^\circ(x_*; v) = \lim_{d \in T^H_{\Omega_{nr}}, d \in \mathcal{D}} f^\circ(x_*; d) \geq 0$. For non-normalized v the result holds since $T^{Cl}_{\Omega_{nr}}(x_*)$ is a cone and the Clarke derivatives are homogeneous in their second arguments. □

Assuming never leaving restoration

Theorem 5.9. *Consider the algorithm 5.1 and let* $a_k = \sum_{i=1}^{m_\mu} \omega_k^i \tilde{d}_k^i$. *Assume that* f *is bounded below. Assume that Restoration is entered and never left.*

(*i*) *Then there exists a refining subsequence.*

(*ii*) *Let* $x_* \in \Omega_{nr}$ *be the limit point of a convergent subsequence of unsuccessful of iterates* $\{x_k\}_K$ *for which* $\lim_{k \in K} \sigma_k = 0$. *Assume that* g *is Lipschitz continuous near* x_*, *and let* $d \in T^H_{\Omega_{nr}}(x_*)$ *a corresponding refining direction. Then either* $g(x_*) = 0$ *(implying* $x_* \in \Omega_r$ *and thus* $x_* \in \Omega$*) or* $g^\circ(x_*; d) \geq 0$.

(*iii*) *Let* $x_* \in \Omega_{nr}$ *be the limit point of a convergent subsequence of unsuccessful of iterates* $\{x_k\}_K$ *for which* $\lim_{k \in K} \sigma_k = 0$. *Assume that* g *and* f *are Lipschitz continuous near* x_*, *and let* $d \in T^H_{\Omega_{nr}}(x_*)$ *a corresponding refining direction such that* $g^\circ(x_*; d) \leq 0$. *Then* $f^\circ(x_*; d) \geq 0$.

(*iv*) *Assume that the interior of the set* $T(x_*)$ *given in (5.11) is non-empty. Let the set of refining directions be dense in* $T(x_*)$. *Then* $f^\circ(x_*, v) \geq 0$ *for all* $v \in T^{Cl}_{\Omega_{nr}}(x_*)$ *such that* $g^\circ(x_*, v) \leq 0$, *and* x_* *is a Clarke stationary point.*

Proof. (*i*) There must exist a refining subsequence K within this call of the restoration, by applying the same argument of the case where one has $g(x_{k+1}) < g(x_k) - \rho(\sigma_k)$ and $g(x_{k+1}) > C\rho(\sigma_k)$ for an infinite subsequence of successful iterations (see the proof of Theorem 5.1). By assumption there exists a subsequence $K' \subseteq K$ such that the sequence $\{a_k / \|a_k\|\}_{k \in K'}$ converges to $d \in T^H_{\Omega_{nr}}(x_*)$ in K' and the sequence $\{\|a_k\| \sigma_k\}_{k \in K'}$ goes to zero in K'. Thus one must have necessarily for k sufficiently large in K', $x_{k+1}^{trial} = x_k + \sigma_k a_k \in \Omega_{nr}$.

(*ii*) Since the iteration $k \in K'$ is unsuccessful in the Restoration,$g(x_k + \sigma_k a_k) \geq g(x_k) - \rho(\sigma_k)$ or $g(x_{k+1}) \leq C\rho(\sigma_k)$, and the proof follows an argument already seen (see the proof of Theorem 5.5).

(*iii*) Since at the unsuccessful iteration $k \in K'$, Restoration is never left, so one has $M(x_k + \sigma_k a_k, \bar{\delta}) \geq M(x_k, \bar{\delta})$, and the proof follows an argument already seen (see the proof of Theorem 5.7).

(*iv*) The same proof as Theorem 5.8. □

Assuming entering leaving restoration an infinite number of times

Theorem 5.10. *Consider the algorithm 5.1 and assume that f is bounded below. Assume that Restoration is entered and left an infinite number of times.*

(*i*) *Then there exists a refining subsequence.*

(*ii*) *Let $x_* \in \Omega_{nr}$ be the limit point of a convergent subsequence of unsuccessful of iterates $\{x_k\}_K$ for which $\lim_{k\in K} \sigma_k = 0$. Assume that g is Lipschitz continuous near x_*, and let $d \in T^H_{\Omega_{nr}}(x_*)$ a corresponding refining direction. Then either $g(x_*) = 0$ (implying $x_* \in \Omega_r$ and thus $x_* \in \Omega$) or $g^\circ(x_*; d) \geq 0$.*

(*iii*) *Let $x_* \in \Omega_{nr}$ be the limit point of a convergent subsequence of unsuccessful of iterates $\{x_k\}_K$ for which $\lim_{k\in K} \sigma_k = 0$. Assume that g and f are Lipschitz continuous near x_*, and let $d \in T^H_{\Omega_{nr}}(x_*)$ a corresponding refining direction such that $g^\circ(x_*; d) \leq 0$. Then $f^\circ(x_*; d) \geq 0$.*

(*iv*) *Assume that the interior of the set $T(x_*)$ given in (5.11) is non-empty. Let the set of refining directions be dense in $T(x_*)$.Then $f^\circ(x_*, v) \geq 0$ for all $v \in T^{Cl}_{\Omega_{nr}}(x_*)$ such that $g^\circ(x_*, v) \leq 0$, and x_* is a Clarke stationary point.*

Proof. (*i*) Let $K_1 \subseteq K$ and $K_2 \subseteq K$ be two subsequences where Restoration is entered and left respectively.

Since the iteration $k \in K_2$ is unsuccessful in the Restoration, one knows that the step size σ_k is reduced and never increased, one then obtains that σ_k tends to zero. By assumption there exists a subsequence $K' \subseteq K_2$ such that the sequence $\{a_k/\|a_k\|\}_{k\in K'}$ converges to $d \in T^H_{\Omega_{nr}}(x_*)$ in K_2 and the sequence $\{\|a_k\|\sigma_k\}_{k\in K'}$ goes to zero in K'.

(*ii*) For all $k \in K'$, one has $g(x_k + \sigma_k a_k) \geq g(x_k) - \rho(\sigma_k)$ or $g(x_{k+1}) \leq C\rho(\sigma_k)$, one concludes that either $g(x_*) = 0$ or $g^\circ(x_*; d) \geq 0$.

(*iii*) For all $k \in K'$, one has $M(x_k + \sigma_k a_k, \bar{\delta}) \geq M(x_k, \bar{\delta})$, and from this we conclude that $f^\circ(x_*; d) \geq 0$ if $g^\circ(x_*; d) \leq 0$.

(*iv*) The same proof as Theorem 5.8. □

5.2 A particularization for only unrelaxable constraints

5.2.1 Algorithm description

In the case where $\Omega_r = \mathbb{R}^n$,i.e. $\Omega = \Omega_{nr}$, the proposed extension of the globally convergent ES to the constrained setting, in Algorithm 5.1, can be simplify to follow a pure feasible approach. In fact, no constraint violation is allowed , i.e. $g(x) = 0$ for all $x \in \Omega$, meaning that the restoration procedure is not worthy anymore. One has only to start feasible and then prevent stepping outside the feasible region by means of an extreme barrier approach. The sufficient decrease condition is applied not to f but to the extreme barrier function f_Ω defined by: These globally convergent ES are described in detail below, in Algorithm 5.3.

5.2.2 Asymptotic results

The step size behavior in this case can be easily derived using the same proof for the unconstrained case (see Lemme 4.1 in Chapter 4). In fact, due to the sufficient decrease condition, one can guarantee that a subsequence of step sizes will converge to zero. From this property and the fact that the step size is significantly reduced (at least by β_2) in unsuccessful iterations, one proves the existence of a refining subsequence.

Asymptotic results when derivatives are unknown

In this section we treat constraints as a pure black box in the sense that no information is assumed known about the constrained set Ω, rather than a yes/no answer to the question whether a given point is feasible. The following theorem is in the vein of those in [18, 166].

Theorem 5.11. *Let $x_* \in \Omega$ be the limit point of a convergent subsequence of unsuccessful of iterates $\{x_k\}_K$ for which $\lim_{k \in K} \sigma_k = 0$. Assume that f is Lipschitz continuous near x_* with constant $\nu > 0$ and that $T_\Omega^H(x) \neq \emptyset$.*

Let $a_k = \sum_{i=1}^{\mu} \omega_k^i \tilde{d}_k^i$. Assume that the directions $\tilde{d}_k^i$'s and the weights ω_k^i's are such that (i) $\sigma_k \|a_k\|$ tends to zero when σ_k does, and (ii) $\rho(\sigma_k)/(\sigma_k \|a_k\|)$ also tends to zero.

Algorithm 5.3: A globally convergent ES for unrelaxable constraints.

Initialization: Choose positive integers λ and μ such that $\lambda \geq \mu$. Select an initial $x_0 \in \Omega$ and evaluate $f(x_0)$. Choose initial step lengths $\sigma_0, \sigma_0^{\text{ES}} > 0$ and initial weights $(\omega_0^1, \ldots, \omega_0^\mu) \in S$. Choose constants $\beta_1, \beta_2, d_{\min}, d_{\max}$ such that $0 < \beta_1 \leq \beta_2 < 1$ and $0 < d_{\min} < d_{\max}$. Select a forcing function $\rho(\cdot)$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. **Offspring Generation:** Compute new sample points $Y_{k+1} = \{y_{k+1}^1, \ldots, y_{k+1}^\lambda\}$ such that
$$y_{k+1}^i \;=\; x_k + \sigma_k \tilde{d}_k^i, \; i = 1, \ldots, \lambda, \tag{5.12}$$
where the directions $\tilde{d}_k^i$'s are computed from the original ES directions d_k^i's (which in turn are drawn from a chosen ES distribution $\mathcal{C}_k$ and scaled if necessary to satisfy $d_{\min} \leq \|d_k^i\| \leq d_{\max}$).

2. **Parent Selection:** Evaluate $f_\Omega(y_{k+1}^i)$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\lambda\}$ by increasing order: $f_\Omega(\tilde{y}_{k+1}^1) \leq \cdots \leq f_\Omega(\tilde{y}_{k+1}^\lambda)$. Select the new parents as the best μ offspring sample points $\{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\mu\}$, and compute their weighted mean
$$x_{k+1}^{trial} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i.$$
Evaluate $f(x_{k+1}^{trial})$.

3. **Imposing Sufficient Decrease:**
If $f_\Omega(x_{k+1}^{trial}) \leq f_\Omega(x_k) - \rho(\sigma_k)$, then consider the iteration successful, set $x_{k+1} = x_{k+1}^{trial}$, and $\sigma_{k+1} \geq \sigma_k$ (for example $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\text{ES}}\}$).
Otherwise, consider the iteration unsuccessful, set $x_{k+1} = x_k$ and $\sigma_{k+1} = \bar{\beta}_k \sigma_k$, with $\bar{\beta}_k \in (\beta_1, \beta_2)$.

4. **ES Updates:** Update the ES step length σ_{k+1}^{ES}, the distribution $\mathcal{C}_k$, and the weights $(\omega_{k+1}^1, \ldots, \omega_{k+1}^\mu) \in S$. Increment k and return to Step 1.

If $d \in T_\Omega^H(x_*)$ is a refining direction associated with $\{a_k/\|a_k\|\}_K$, then $f^\circ(x_*; d) \geq 0$.

If the set of refining directions associated with $\{a_k/\|a_k\|\}_K$ is dense in the unit sphere, then x_* is a Clarke stationary point.

Proof. To prove the first part one can use the same proof as in Theorem 4.3.

For the second part, we first conclude from the density of the refining directions on the unit sphere and the continuity of $f^\circ(x_*; \cdot)$ in $T_\Omega^H(x_*)$, that $f^\circ(x_*; d) \geq 0$ for all $d \in T_\Omega^H(x_*)$. Finally, we conclude that $f^\circ(x_*; v) = \lim_{d \in T_\Omega^H(x_*), d \to v} f^\circ(x_*; d) \geq 0$ for all $v \in T_\Omega(x_*)$. □

Asymptotic results when derivatives are known

Although the approach analyzed in Subsection 5.2.2 can in principle be applied to any type of constraints, it is obviously more appropriate to the case where one cannot compute the derivatives of the functions algebraically defining the constraints.

Now we consider the case where we can compute tangent cones at points on the boundary of the feasible set Ω. This is the case whenever Ω is defined by $\{x \in \mathbb{R}^n : \; c_i(x) \leq 0, \; i \in \mathcal{I}\}$ and the derivatives of the functions c_i are known. Two particular cases that appear frequently in practice are bound and linear constraints.

For theoretical purposes, let ϵ be a positive scalar and k_0 a positive integer. Let us also denote by T_{Ω,ϵ,k_0} the union of all Clarke tangent cones $T_\Omega(y)$ for all points y at the boundary of Ω such that $\|y - x_k\| \leq \epsilon$ for all $k \geq k_0$.

Theorem 5.12. *Let $x_* \in \Omega$ be the limit point of a convergent subsequence of unsuccessful of iterates $\{x_k\}_K$ for which $\lim_{k\in K} \sigma_k = 0$. Assume that f is Lipschitz continuous near x_* with constant $\nu > 0$ and that $T_\Omega^H(x) \neq \emptyset$.*

Let $a_k = \sum_{i=1}^{\mu} \omega_k^i \tilde{d}_k^i$. Assume that the directions $\tilde{d}_k^i$'s and the weights ω_k^i's are such that (i) $\sigma_k \|a_k\|$ tends to zero when σ_k does, and (ii) $\rho(\sigma_k)/(\sigma_k\|a_k\|)$ also tends to zero.

If $d \in T_\Omega^H(x_*)$ is a refining direction associated with $\{a_k/\|a_k\|\}_K$, then $f^\circ(x_*; d) \geq 0$.

If the set of refining directions associated with $\{a_k/\|a_k\|\}_K$ is dense in the intersection of T_{Ω,ϵ,k_0} with the unit sphere (for some $\epsilon > 0$ and positive integer k_0), then x_* is a Clarke stationary point.

Proof. It has already been shown in Theorem 5.11 that if $d \in T_\Omega^H(x_*)$ is a refining direction associated with $\{a_k/\|a_k\|\}_K$, then $f^\circ(x_*; d) \geq 0$.

The rest of the proof results from the fact that the Clarke tangent cone $T_\Omega(x_*)$ is contained in T_{Ω,ϵ,k_0} for any limit point x_* of a subsequence of iterates (and in particular for the subsequence K in the statement of the theorem). Thus, $f^\circ(x_*; v) = \lim_{d\in T_\Omega^H(x_*), d\to v} f^\circ(x_*; d) \geq 0$ for all $v \in T_\Omega(x_*)$. □

5.2.3 Implementation choices

In this subsection, we address linearly unrelaxable constrained problems of the form (5.1) where Ω_{nr} is defined as $\{x \in \mathbb{R}^n : Cx \leq d\}$, $C \in \mathbb{R}^{m\times n}$, and $d \in \mathbb{R}^m$, for some positive integer m.

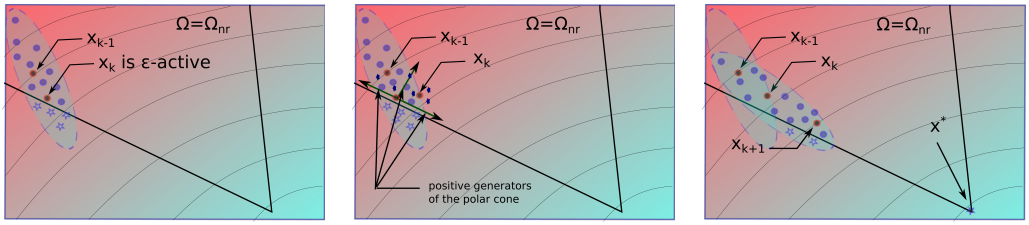
Approach based on extreme barrier and the inclusion of positive generators

A known technique for handling unrelaxable constraints with known constrained derivative information is based on computing sets of positive generators for appropriate tangent cones. By a set of positive generators of a convex cone, it is meant a set of vectors that spans the cone with nonnegative coefficients. A difficulty when using integer/rational lattices as a globalization strategy (for driving the step size parameter to zero) in the nonlinear case is that the positive generators of the tangent cones in consideration would lack of rationality. What makes it possible to derive a result like Theorem 5.12 valid for nonlinear constraints is the combination of (i) a sufficient decrease condition for accepting new iterates (which took care of the need to drive the step size parameter to zero) with (ii) the dense generation of the directions in tangent cones (which prevents stagnation at boundary points). We note that there are a number of globally convergent *hybrid* approaches using penalty or augmented Lagrangian functions (see [118]) or filter techniques (see [19]), but without attempting to compute positive generators of the appropriated tangent cones related to the nonlinear part of the constraints.

In the literature of direct-search methods (of directional type) for constraints, one finds approaches specifically developed for the bound or linear constrained cases (see [77, 108, 110, 117]), where positive generators of the appropriated tangent cones are computed and used for what is called polling (i.e. for evaluating the objective function at points of the form $x_k+\sigma_k d$, where d is a positive generator). Although we also address constraints of that type in this work, we do not want to resort our poll of directions completely to such positive generators as that would not allow to take advantage of the ES random mechanism (Theorem 5.12 would however provide a possible theoretical coverage for such an approach). Instead, we propose to modify the set of directions generated by ES to include positive generators of appropriate tangent cones. The details will be given in the rest of the current section.

The point to make here is that the global convergence result of Theorem 5.11 remains valid as long as the set $\{\tilde{d}_k^i, i = 1, \ldots, \lambda\}$ still verifies Assumptions (i) and (ii). Assumption (i) is trivially satisfied as long as all the positive generators $\tilde{d}_k^i$ are bounded above in norm, which is explicit in the algorithm when $\tilde{d}_k^i = d_k^i$ is an ES randomly generated direction (and can be trivially imposed if $\tilde{d}_k^i$ is a positive generator). The satisfaction of Assumption (ii) is met, for instance, if a_k is bounded below in norm. That in turn depends on the calculation of the all the $\tilde{d}_k^i$'s and on the choice of the weights ω_k^i's, but can always be achieved in the limit case where one weight is set to one and the others to zero.

In this approach we form the set of directions $\{\tilde{d}_k^i\}$ by first replacing some of the ES randomly generated directions d_k^i, whenever the current iterate is closer to the boundary of the feasible region, by positive generators of an appropriated tangent cone (see Figure 5.1).



(a) First step (b) Second step (c) Third step

FIGURE 5.1: A 2-D illustration of the barrier approach to handle linearly constrained problems using a positive generators of the polar cone of the ϵ-active constraints. Figure (5.1(a)) outlines the detection of an ϵ-active mean parent point, while Figures (5.1(b)) and (5.1(c)) show the restoration process to conform the offspring distribution to the local geometry. The ellipses show the level sets of the objective function.

More specifically, at the current iterate x_k, given $\epsilon_k > 0$, we first identify the ϵ_k-active constraints $I_k = \{i \in \{1, \ldots, m\} : c_i x_k - d_i \geq -\epsilon_k\}$, where c_i denotes the i-th line of C, and then represent by $C_k \in \mathbb{R}^{|I_k| \times n}$ the submatrix of C formed by the rows associated with the ϵ_k-active constraints. The directions to be considered for inclusion are the positive generators D_k of the tangent cone formed at a point where the active constraints are those in I_k. We choose ϵ_k to be $\mathcal{O}(\sigma_k)$ as in [110] (to avoid considering all positive generators for all tangent cones for all $\epsilon \in [0, \epsilon_*]$ where $\epsilon_* > 0$ is independently of the iteration counter as proposed in [117]). We then use the following algorithm from [165] to compute the set D_k of positive generators for corresponding tangent cone (in turn inspired by the work in [4, 117]). Basically, the idea of this algorithm is to dynamically decrease ϵ_k in the search for a set of positive generators of a tangent cone corresponding to a full row rank matrix C_k.

The final set of directions $\{\tilde{d}_k^i, i = 1, \ldots, \lambda\}$ is then formed by selecting among $\{d_k^i, i = 1, \ldots, \lambda\} \cup D_k$ those that lead to the best objective function value at the points $x_k + \sigma_k d$ with $d \in \{d_k^i, i = 1, \ldots, \lambda\} \cup D_k$.

Approach based on projecting onto the feasible region

The second approach to deal with the unrelaxable constraints is based on projecting onto the feasible domain all the generated sampled points $x_k + \sigma_k d_k^i$, and then taking instead $\Phi_\Omega(x_k + \sigma_k d_k^i)$. We note that projecting onto the feasible region in the context of derivative-free optimization has been already advocated in [120].

Algorithm 5.4: Calculating the positive generators D_k.

Initialization: Choose $\epsilon_k = \min(0.1, 10\sigma_k)$ and $\epsilon_{limit} = \min(0.1, \epsilon_k^2)$.

While $\epsilon > \epsilon_{limit}$

1. Construct the matrix C_k.
2. If $0 < \dim(C_k) < n$ and C_k is full rank, then
   a. Compute a QR factorization of the matrix $C_k^\top$.
   b. Let $Z_k = QR^{-\top}$, $Y_k = I - Z_k C_k$, and stop with $D_k = [\, Z_k \;\; -Z_k \;\; Y_k \;\; -Y_k \,]$.
3. If $\dim(C_k) = 0$, then stop (and return $D_k = [\;\;]$), else $\epsilon_k = \epsilon_k/2$.

End While.

This procedure is however equivalent to consider

$$\tilde{d}_k^i \;=\; \frac{\Phi_\Omega(x_k + \sigma_k d_k^i) - x_k}{\sigma_k}$$

in the framework of Algorithm 5.3. By substituting all the infeasible generated sampled points by their projections one also conforms the distribution of the offspring to the local geometry of the constraints. Unlike in the first approach, one does need here to make use of the extreme barrier function and thus its presence in Steps 2 and 3 of Algorithm 5.3 is innocuous.

Again, the global convergence results remains valid as long as the set $\{\tilde{d}_k^i, i = 1, \ldots, \lambda\}$ still verifies Assumptions (i) and (ii). If we look at Assumption (i), one sees that

$$\sigma_k \|a_k\| \;=\; \left\| \sum_{i=1}^{\mu} \omega_k^i [\Phi_\Omega(x_k + \sigma_k d_k^i) - x_k] \right\| \;\leq\; \sigma_k \sum_{i=1}^{\mu} \omega_k^i L_{\Phi_\Omega} \|d_k^i\|,$$

since $x_k = \Phi_\Omega(x_k)$, where we assumed that the projection mapping Φ_Ω is Lipschitz continuous with constant $L_{\Phi_\Omega} > 0$. Since the d_k^i's are bounded above in norm, one concludes that $\sigma_k \|a_k\|$ does indeed tend to zero. Note that the projection Φ_Ω is Lipschitz continuous when defined in the best approximation sense using some norm or distance (being the constant L_{Φ_Ω} equal to 1 in the Euclidean/ℓ_2 case). The satisfaction of Assumption (ii) is achieved if a_k is bounded below in norm and similar considerations as in the previous approach apply here too.

For this approach based on projecting onto the feasible region one needs to define the projection mapping Φ_Ω. Given a norm $\|\cdot\|$ and a nonempty closed, convex set Ω, the

mapping Φ_Ω can be defined as:

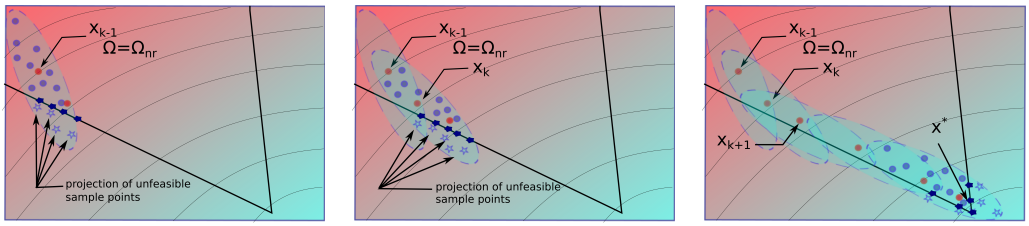
$$\Phi_\Omega(x) \;=\; \arg\min\{\|z - x\| : z \in \Omega\}. \tag{5.13}$$

For purely bound constrained problems, when $\Omega = \{x \in \mathbb{R}^n : l \leq x \leq u\}$, we will use the ℓ_2-norm since it reduces to a trivial computation. In fact, in the Euclidean case, the projection (5.13) is simply given by (for $i = 1, \ldots, n$)

$$[\Phi_\Omega(x)]_i \;=\; \begin{cases} l_i & \text{if } x_i < l_i, \\ u_i & \text{if } x_i > u_i, \\ x_i & \text{otherwise.} \end{cases}$$

For general linearly constrained problems, the Euclidean/ℓ_2 projection (5.13) reduces to the solution of a QP problem with inequality constraints. We will rather use the projection (5.13) when the norm is the ℓ_1 one as its evaluation requires instead the solution of an LP problem.

Another possibility would be to damp the step and allow the longest displacement along each direction, in other words to compute for each direction $\tilde{d}_k^i$ the largest $\alpha_k^i \in (0, 1]$ such that $y_{k+1}^i = x_k + \alpha_k^i(\sigma_k \tilde{d}_k^i) \in \Omega$. Although such a projection does not require the solution of any auxiliary problem, it depends on the iteration counter and, furthermore, it did not lead to better overall results when compared to the ℓ_1 one. Figure 5.2 depicts a 2D illustration of the projection approach.



(a) First step (b) Second step (c) Last step

FIGURE 5.2: An illustration of the projection approach to handle linearly constrained problems. The figure (5.2(a)) outlines the projection of the unfeasible sample points. Figures (5.2(b)) and (5.2(c)) show the adaptation of the distribution of the offspring candidate solution to the constraints local geometry.

5.3 Numerical experiments

5.3.1 Unrelaxable constraints

As a first test scenario, we have evaluated the performance of Algorithm 5.3 proposed for only unrelaxable constraints and under the choices described in Section 5.2, using different solvers, different comparison procedures, and a large collection of problems of more than 200 bound and linearly constrained instances.

The solver related to the barrier approach will be called ES-LC-B standing for an **E**volution **S**trategy to handle **L**inear **C**onstraints using a **B**arrier approach. The solver related to the projection approach will be called ES-LC-P standing for an **E**volution **S**trategy to handle **L**inear **C**onstraints using a **P**rojection approach. We were mainly interested in observing the efficiency and the robustness of our algorithms.

5.3.1.1 Solvers tested

The solvers used for our numerical comparisons were BCDFO, CMA-ES, MCS, and PSWARM:

- BCDFO [72], Matlab version of Oct. 25, 2011. BCDFO is a local quadratic interpolation-based trust-region algorithm for bound constrained problems.
- CMA-ES (Covariance Matrix Adaptation Evolution Strategy) for bound constrained optimization, 3.61.beta Matlab version [78, 83]. This constrained version adds to the objective function a penalization term measuring the distance between the current point and its ℓ_2-projection onto the feasible region.
- MCS [93] for bound constrained optimization, 2.0 Matlab version. MCS does a multilevel coordinate search that balances global and local search (the latter using quadratic interpolation).
- PSWARM, the same Matlab version used in [164, 165]. PSWARM implements a polling type direct-search algorithm enhanced by a search step based on swarm optimization for global search. Available for general linear constraints.

In the comparative study published in [145], MCS was among the best solvers in terms of both efficiency and robustness. Among the stochastic solvers tested, CMA-ES and PSWARM have appeared well ranked. BCDFO was developed after this study was carried out but it was shown to perform very well [72].

The default parameters of these four solvers were kept untouched, except the starting point, the initial step size, and the maximal budget, which were chosen the same for all of them including ours.

5.3.1.2 Algorithmic choices

The parameter choices of Algorithm 5.3 followed those in Chapter 4 for unconstrained optimization. The values of λ and μ and of the initial weights are those of CMA-ES for unconstrained optimization (see [78]): $\lambda = 4 + \text{floor}(3\log(n))$, $\mu = \text{floor}(\lambda/2)$, where $\text{floor}(\cdot)$ rounds to the nearest integer, and $\omega_0^i = a_i/(a_1 + \cdots + a_\mu)$, $a_i = \log(\lambda/2 + 1/2) - \log(i)$, $i = 1, \ldots, \mu$. The choices of the distribution $\mathcal{C}_k$ and of the update of σ_k^{ES} also followed CMA-ES for unconstrained optimization (see [78]). The forcing function selected was $\rho(\sigma) = 10^{-4}\sigma^2$. To reduce the step length in unsuccessful iterations we used $\sigma_{k+1} = 0.9\sigma_k$ which corresponds to setting $\beta_1 = \beta_2 = 0.9$. In successful iterations we set $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\text{CMA-ES}}\}$ (with $\sigma_k^{\text{CMA-ES}}$ the CMA step size used in ES). The directions d_k^i, $i = 1, \ldots, \lambda$, were scaled if necessary to obey the safeguards $d_{\min} \leq \|d_k^i\| \leq d_{\max}$, with $d_{\min} = 10^{-10}$ and $d_{\max} = 10^{10}$.

The initial step size is estimated using only the bound constraints, as in [165]: If there is a pair of finite lower and upper bounds for a variable, then σ_0 is set to half of the minimum of such distances, otherwise $\sigma_0 = 20$. The starting point is set to what is suggested in the problem file (or to the origin when there is no suggestion), if such a choice is feasible. When such a choice is not feasible (the majority of the cases), the starting point is the center of the maximum volume ellipsoid inscribed in the feasible region. As in [165], for computing such an ellipsoid we used the software implementation in [170].

In Algorithm ES-LC-P, for bound constrained problems we will use the ℓ_2-projection (as it is trivial to evaluate) and in the case of general linear constraints we will use the ℓ_1-projection (as it reduces to the solution of an LP) where we use the Matlab `linprog` routine.

5.3.1.3 Test problems

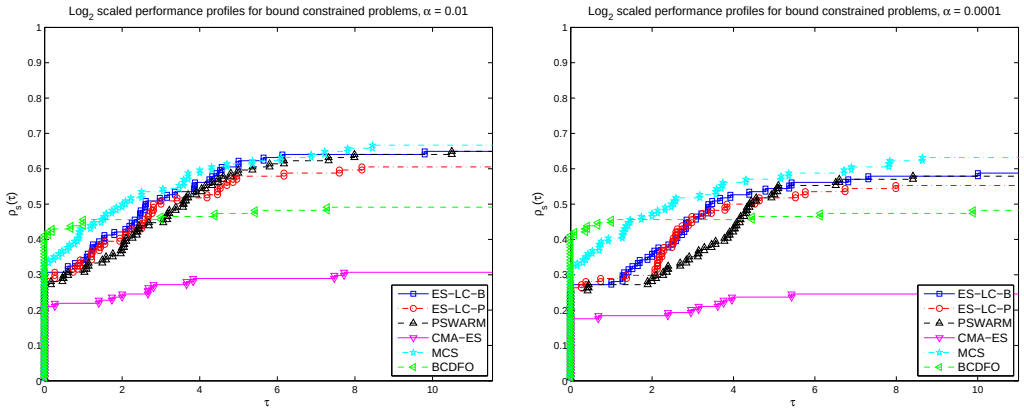
Our test problem set $\mathcal{P}$ is taken from the one used in [164, 165] to compare PSWARM with other solvers and where the problems were collected from known non-linear programming testing collections. The problems are coded in AMPL and divided into two groups. The first group includes only pure bound constraints problems and it gathers 114 problems essentially from [7, 95, 119, 124]. The second group includes 107 general

linear constrained problems, collected essentially from [1, 2]. All the solvers were thus interfaced to AMPL. Relatively to the list of test problems reported in [164, 165] we have excluded the bounded constrained problems `lms1a`, `lms1b`, `lms2`, `lms3`, `lms5` due to library linkage and the linearly constrained problems `antenna2`, `powell20` for which none of the solvers were able to find a feasible starting point. The problems and there dimension distribution are listed in Appendix B.

5.3.1.4 Comparison results

For our numerical experiments, we used a maximal computational budget consisting of 1500 function evaluations. Again as in the previous chapter, we choose to work with data and performance profiles to assist the performance of the tested solvers (see Section 4.2.3). When the solver is stochastic we plot the average profile over the number of runs. The complete comparaison result is given in Appendix B.

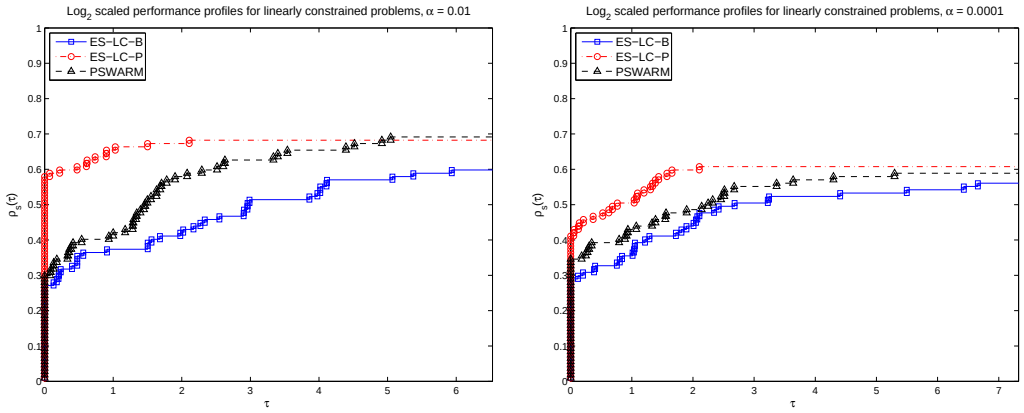
The performance profiles are used to quantify the ability of the tested solvers to approach the global minimum of a given problem. For the convergence test, we use the global minimum when it is known, otherwise it is chosen as the best objective function value found by all the tested solvers using an extremely large computational budget (a number of function evaluations equal to 500000). Thus, in such a case it makes more sense not to select the accuracy level too small, and our tests were performed with $\alpha = 10^{-2}, 10^{-4}$. The plots in Figure 5.3 and Figure 5.4 outline the performance profile results. The left side of these Figures gives the percentage of the test problems, out of the problem tested, for which an algorithm is more successful (efficiency). The right side represents a measure of an algorithm's robustness.



(a) Accuracy level of 10^{-2}.

(b) Accuracy level of 10^{-4}.

FIGURE 5.3: Performance profiles for 114 bound constrained problems (average objective function values for 10 runs).



(a) Accuracy level of 10^{-2}.

(b) Accuracy level of 10^{-4}.

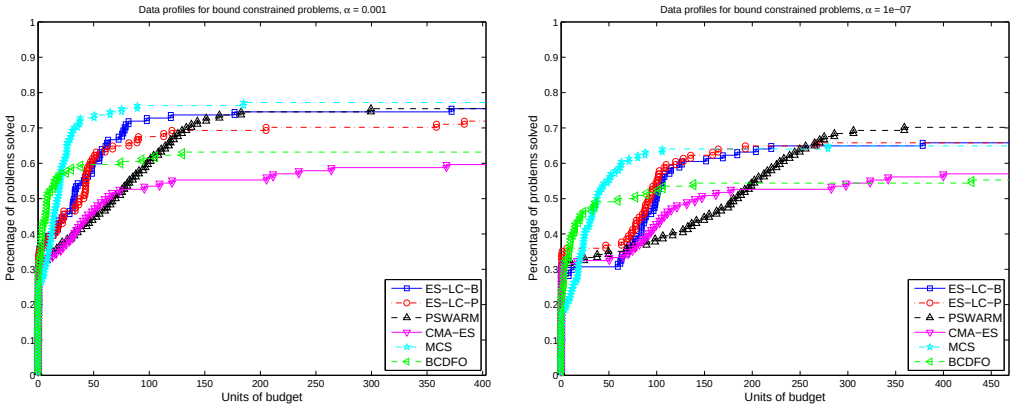
FIGURE 5.4: Performance profiles for 107 general linearly constrained problems (average objective function values for 10 runs).

For bound constrained problems (Figure 5.3), one can see that BCDFO and MCS are better than the other solvers in terms of efficiency. ES-LC-B appears to be the most efficient stochastic solver. In terms of robustness, ES-LC-B, ES-LC-P, PSWARM and MCS show very good performance. When it comes to BCDFO solver, it looses a lot in terms of robustness. CMA-ES performs the worst in both efficiency and robustness. The good efficiency performance of MCS and BCDFO is not surprising since they are based on interpolation models and most of the objective functions tested are smooth. Moreover, both solvers are specifically designed to solve bound constraints problems.

For general linear constrained problems (Figure 5.4), the projection approach (ES-LC-P) performs, in both efficiency and robustness, better than PSWARM. However, the barrier approach (ES-LC-B) showed the worst profile in terms of efficiency.

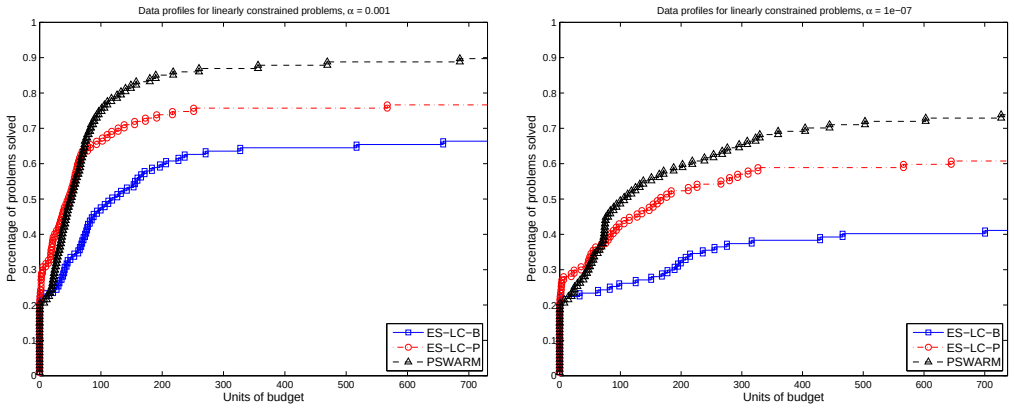
As a second scenario test, we are primarily interested in the behavior of the algorithms for problems where the evaluation of the objective function is expensive using data profiles. As for the levels of accuracy, we chose two values, $\alpha = 10^{-3}$ and $\alpha = 10^{-7}$. Since the average of the best objective value f_L is chosen as the average best value found by all solvers considered, but under a relatively low maximal computational budget, it makes no sense then to consider a high accuracy level (less than 10^{-7}).

For bound constrained problems (Figure 5.5), the solvers ES-LC-P, ES-LC-B, PSWARM and MCS perform well, with an advantage of MCS for small budget. BCDFO and CMA-ES are less competitive compared to the other solvers. Regarding general linear constraints (Figure 5.6), ES-LC-P and PSWARM perform in the same way. ES-LC-B showed the worst performances for this class of problems.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 5.5: Data profiles for 114 bound constrained problems (average objective function values for 10 runs).



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 5.6: Data profiles for 107 general linearly constrained problems (average objective function values for 10 runs).

5.3.2 Relaxable and unrelaxable constraints

The objective of this chapter is to propose a new approach handling both relaxable and unrelaxable constraints. On the previous works, only the progressive barrier [19] is able to handle both constraints. Thus, to quantify the efficency of the merit approach we compare our solver with the direct search method MADS where the progressive barrier approach is been implemented. We used the implementation given in the NOMAD package [3, 16, 116], version 3.6.1 (C++ version linked to Matlab via a mex interface), where we enabled the option `DISABLE MODELS`, meaning that no modeling is used in MADS. The models are disabled since our solvers at this stage are not using any modeling to speed up the convergence. The reader is referred to Chapter 6 for model incorporation in direct-search methods.

5.3.2.1 Test problems

Our test set is the one used in [87, 90, 111, 123] and comprises 13 well-known test problems `PrG1`–`PrG13`. The problems `PrG2`,`PrG3` and `PrG8` are maximization problems, so they will be solved by converting them to minimization problems. The characteristics of those test problems are diverse and tend to cover many kind of difficulties that constrained global optimization problems face. In addition to such problems, we add three other engineering optimization problems [45, 87]: `PrP` the pressure vessel design problem, `PrT` the tension-compression string problem, and `PrW` the welded beam design problem. The following table contains the distribution of the dimension n, the number of the constraints m (in addition to the bounds), and the best know optimal value f_{opt} for the chosen problems:

Name	n	m	f_{opt}
`PrG1`	13	9	-15
`PrG2`	20	2	-0.803619
`PrG3`[a]	20	1	-1
`PrG4`	5	6	-30665.5
`PrG5`[a]	4	5	5126.5
`PrG6`	2	2	-6961.81
`PrG7`	10	8	24.3062
`PrG8`	2	2	-0.095825
`PrG9`	7	4	680.63
`PrG10`	8	6	7049.33
`PrG11`[a]	2	1	0.75
`PrG12`	3	1	-1
`PrG13`	5	3	0.0539498
`PrP`	4	3	5868.76
`PrT`	3	4	0.0126653
`PrW`[a]	4	6	1.725

[a]Problems contain equality constraints. When a constraint is of the form $c_i^e(x) = 0$, we use the following relaxed constraint instead $c_i(x) = |c_i^e(x)| \leq 10^{-4}$.

5.3.2.2 Test strategy

To investigate the advantages and drawbacks of the merit approach over the classical extreme barrier approach. The initial step size is estimated using only the bound constraints: If there is a pair of finite lower and upper bounds for a variable, then σ_0 is set to the half of the minimum of such distances, otherwise $\sigma_0 = 1$. The starting point

is the same for all solvers and set to $(LB + UB)/2$ when the bounds LB and UB are given, otherwise it is chosen randomly in the search space.

We first have made our test by considering that all the constraints as unrelaxable, thus only the extreme barrier approach given by Algorithm 5.3. The solver related to the barrier approach will be called ES-EB standing for an **E**volution **S**trategy using the **E**xtreme **B**arrier approach. ES-EB will be compared to MADS where we disable the progressive barrier and enable the extreme one, the related solver will be called MADS-EB. However unlike the merit approach or the progressive barrier where starting from an infeasible point is possible, the extreme barrier needs a feasible point. For infeasible starting points, we use for ES-EB the same strategy implemented in MADS-EB [19] where a two-phases extreme barrier approach is used as follows. During a first phase, one neglects the objective function and tries to generate a feasible point (possibly with a large objective function value). Finding a feasible point can be guarantied by finding a global minimizer to the following optimization problem:

$$\begin{aligned} \min \quad & g(x) \\ \text{s.t.} \quad & x \in \mathbb{R}^n, \end{aligned} \tag{5.14}$$

where g is the constraints violation function defined by (5.4). Once a feasible point is found, a second phase is launched to find a better point in terms of the objective function.

In the second test scenario, we consider that all the constraints are relaxable except the bounds. In this case, the merit function (MF) and the progressive approaches (PB) are respectively enabled, the related solvers will be called ES-MF and MADS-PB respectively.

5.3.2.3 Numerical results

Tables 5.1 and 5.2, report results for both ES-EB and MADS-ES using a maximal budget of 2000 and 20000, respectively. For each problem, we display the optimal objective value found the tested solver. In order to show more details concerning the convergence cost, the number of function evaluations consumed by the tested solved is also reported. To be more precise, we display for the ES-EB the number of objective function evaluations $\#f$, used for in the second phase, as well as the number of the constraints violation evaluation $\#g$, used in the first phase. MADS-EB evaluates the whole problem (objective function and constraints) on the same time as black box problem, thus we report a global number of evaluations of the black box including both the objective constraints function evaluations $\#(f, g)$.

For a maximum number of function evaluations of 2000 (Table 5.1), both solvers have no difficulty finding a feasible point when the starting point is infeasible. However, the solvers are not able to find the global optimum of all the problems. For the ones that converge (i.e. `PrG3`, `PrG8`,`PrG11`,`PrG12` and `PrT`), ES-EB tends to converge to a better solutions compared to MADS-EB for the problems `PrG3` and `PrT`. MADS-EB and ES-EB converge to the same optimum for `PrG8`, `PrG11` and `PrG12` with a slightly advantage to the MADS-EB in terms of the convergence cost.

Name	Best known	ES-EB			MADS-EB	
	f_{opt}	$f(x_*)$	$\#f$	$\#g$	$f(x_*)$	$\#(f,g)$
`PrG1`	-15	-14.2698	2000	1027	-7.82754	2000
`PrG2`	-0.803619	-0.229242	2000	0	-0.206025	2000
`PrG3`	-1	$\mathbf{-0.0383156}$	85	2000	$-6.36481e-233$	1310
`PrG4`	-30665.5	-30498.1	2000	331	-30658.3	2000
`PrG5`	5126.5	5976.79	5	1648	5361.97	2000
`PrG6`	-6961.81	$\mathbf{-6961.81}$	985	186	$\mathbf{-6961.81}$	2000
`PrG7`	24.3062	37.1668	2000	841	27.9464	2000
`PrG8`	-0.095825	$\mathbf{-0.095825}$	483	186	$\mathbf{-0.095825}$	343
`PrG9`	680.63	682.643	2000	0	681.667	2000
`PrG10`	7049.33	16688.4	2000	885	7953.48	2000
`PrG11`	0.75	**0.99998**	197	0	**0.9998**	193
`PrG12`	-1	$\mathbf{-1}$	248	0	$\mathbf{-1}$	173
`PrG13`	0.0539498	2.7922	2	892	0.999626	2000
`PrP`	5868.76	7027.49	2000	253	5916.24	2000
`PrT`	0.0126653	**0.0135759**	667	477	0.0161624	936
`PrW`	1.725	2.41206	2000	487	4.63432	2000

TABLE 5.1: Comparison results for the extreme barrier approach using a maximal budget of 2000 .

For a large maximal number of function evaluation of 20000 (Table 5.2), ES-EB and MADS-EB achieve convergence to a stationary point for most of the problems, except for MADS-EB which does not converge for the problems `PrG10` and `PrG13`. The advantage of ES-EB over MADS-EB is getting clearer in the large budget case. In fact, ES-EB reaches a better solution (if not the global ones) than MADS-EB for 50% of the problems (i.e. `PrG1`, `PrG2`, `PrG3`, `PrG4`, `PrG7`, `PrG9` , `PrT` and `PrW`). Both solvers converge to the same value for 25% of the problems (i.e. `PrG6`, `PrG8` , `PrG11` and `PrG12`). MADS-EB is shown to be better on the 25% problems left. ES-EB is showing bad performance on the problems `PrG3`, `PrG5` and `PrG13` since the constraints are mainly the equality ones. Thus ES-EB has not a lot of freedom to sample point in the search space. In fact, after generating a feasible point, ES-EB is not able to progress towards better region. As far as the constraints violation will be allowed, we expect that our ES will be able to improve the quality of the found solution.

Name	Best known	ES-EB			MADS-EB	
	f_{opt}	$f(x_*)$	$\#f$	$\#g$	$f(x_*)$	$\#(f,g)$
PrG1	-15	$\mathbf{-14.9999}$	17259	1027	-7.82761	10093
PrG2	-0.803619	$\mathbf{-0.229242}$	3290	0	-0.206864	11027
PrG3	-1	$\mathbf{-0.0383278}$	227	2584	$-6.36481e-233$	1310
PrG4	-30665.5	$\mathbf{-30665.4}$	5598	331	-30664.9	6666
PrG5	5126.5	5976.79	5	1648	**5361.97**	2166
PrG6	-6961.81	$\mathbf{-6961.81}$	985	186	$\mathbf{-6961.81}$	2027
PrG7	24.3062	**24.4533**	17491	841	27.8811	5010
PrG8	-0.095825	$\mathbf{-0.095825}$	483	186	$\mathbf{-0.095825}$	343
PrG9	680.63	**680.63**	18969	0	681.301	3443
PrG10	7049.33	16220.9	6987	885	**7933.19**	20000
PrG11	0.75	**0.99998**	197	0	**0.9998**	193
PrG12	-1	$\mathbf{-1}$	248	0	$\mathbf{-1}$	173
PrG13	0.0539498	2.7922	2	892	**0.997151**	20000
PrP	5868.76	6044.93	3764	253	**5916.23**	4219
PrT	0.0126653	**0.0135759**	667	477	0.0161624	936
PrW	1.725	**2.21999**	5159	487	4.49103	3248

TABLE 5.2: Comparison results for the extreme barrier approach using a maximal budget of 20000 .

Tables 5.3 and 5.4, report results for both ES-MF and MADS-PB using a maximal budget of 2000 and 20000, respectively. For each problem, we display the optimal objective value found by the solver $f(x_*)$, the associated constrained violation $g(x_*)$, and the number of objective function evaluations $\#f$ needed to reach x_*. When a solver returns a flag error or encounters an internal problem, we display '$-$' instead of the values of $f(x_*)$ and $g(x_*)$.

Table 5.3 gives the obtained results for a maximal budget of 2000 function evaluations. Except four problems both solvers are not able to converge for the regarded budget. Again, our solver ES-EM is shown to be more global than MADS-PB on the tested problems.

For a large maximal number of function evaluation of 20000 (Table 5.4), ES-MF and MADS-PB achieve convergence to a stationary point for most of the problems, except two problems PrG10 and PrG13, where MADS-PB is requiring more function evaluations. The advantage of ES-MF over MADS-PB is evident compared to the small budget case. In fact, one can observe that ES-MF reaches better solutions (if not the global ones) than MADS-EB for ten of the sixteen tested problems with a 10^{-5} as constraints violation tolerance (i.e. PrG1, PrG2, PrG3, PrG5, PrG7, PrG9 ,PrG11 , PrG13, PrT and PrW). Both solvers converge to the same value for two problems PrG8 and PrG12. MADS-EB is shown to be better on the following four problems: PrG3, PrG4, PrG6 and PrG10.

Name	Best known	ES-MF			MADS-PB		
	f_{opt}	$f(x_*)$	$\#f$	$g(x_*)$	$f(x_*)$	$\#f$	$g(x_*)$
PrG1	-15	$\mathbf{-12.8618}$	2000	$7.3e-07$	-8.99982	2000	0
PrG2	-0.803619	-0.27127	2000	0	-0.214849	2000	0
PrG3	-1	-0.00126411	2000	$1.8e-09$	$-3.69017e-06$	1310	0
PrG4	-30665.5	-31070.1	2000	0.29	-30665.4	2000	0
PrG5	5126.5	5628.47	2000	0.0029	5237.9	2000	0.26
PrG6	-6961.81	-7905.27	2000	0.48	-6961.81	2000	0
PrG7	24.3062	29.8256	2000	0	33.2519	2000	0
PrG8	-0.095825	$\mathbf{-0.095825}$	274	0	$\mathbf{-0.095825}$	343	0
PrG9	680.63	681.699	2000	0	680.904	2000	0
PrG10	7049.33	9647.75	2000	3	6198.97	2000	0.023
PrG11	0.75	0.74974	2000	$2.6e-08$	0.9998	193	0
PrG12	-1	$\mathbf{-1}$	241	0	$\mathbf{-1}$	173	0
PrG13	0.0539498	1.30217	2000	0.051	0.998843	2000	0
PrP	5868.76	61515.5	2000	0.0013	7542.07	2000	0
PrT	0.0126653	**0.0132653**	1572	$3.2e-10$	–	936	–
PrW	1.725	2.93783	2000	0	3.74259	2000	0

TABLE 5.3: Comparison results for the merit approach and the progressive barrier one using a maximal budget of 2000 .

If one has a tolerance of 10^{-5} on the constraints violation, then both solvers are seen to converge to an unfeasible solution for the problem PrG10. Under such tolerance, MADS-PB is also converging to an unfeasible solution for the problem PrG5 for which ES-MF was able to reach the global minimum.

Name	Best known	ES-MF			MADS-PB		
	f_{opt}	$f(x_*)$	$\#f$	$g(x_*)$	$f(x_*)$	$\#f$	$g(x_*)$
PrG1	-15	$\mathbf{-14.954}$	11222	$2.4e-06$	-8.99999	10093	0
PrG2	-0.803619	$\mathbf{-0.27127}$	3550	0	-0.226599	11027	0
PrG3	-1	$\mathbf{-0.826461}$	17735	$1.8e-05$	-0.00413072	1310	0
PrG4	-30665.5	-29730.1	6810	0	$\mathbf{-30665.4}$	6666	0
PrG5	5126.5	**5126.5**	3970	0	5240.95	2166	0.008
PrG6	-6961.81	-7020.87	3641	0.0084	$\mathbf{-6961.81}$	2027	0
PrG7	24.3062	**24.7564**	11402	0	27.1991	5010	0
PrG8	-0.095825	$\mathbf{-0.095825}$	274	0	$\mathbf{-0.095825}$	343	0
PrG9	680.63	**680.629**	7563	$4.2e-07$	**680.799**	3443	0
PrG10	7049.33	10126.8	8103	0.067	6192.82	20000	0.021
PrG11	0.75	**0.749487**	2807	$1.7e-07$	0.9998	193	0
PrG12	-1	$\mathbf{-1}$	241	0	$\mathbf{-1}$	173	0
PrG13	0.0539498	**0.45309**	8944	$4.5e-09$	0.996284	20000	0
PrP	5868.76	74239	5405	0	**7542.07**	4219	0
PrT	0.0126653	**0.0132653**	1572	$3.2e-10$	–	936	–
PrW	1.725	**2.22794**	8686	0	3.7413	3248	0

TABLE 5.4: Comparison results for the merit approach and the progressive barrier one using a maximal budget of 20000 .

For problems with equality constraints the feasible region is very small, unlike the ES-EB which was not able to make progress after finding a feasible point, the merit approach ES-MF makes more progress by allowing the constraints violation. One can observe that for the three problems with equality constraints (i.e. `PrG3`, `PrG5` and `PrG13`), ES-EB was able to explore only few feasible point contrary to ES-MF. Such freedom during the exploration leads ES-MF to outperform all the other solvers ,i.e. meaning ES-EB, MADS-EB and MADS-PB for the problems tested with equality constraints.

5.4 Conclusions

Motivated by the fact that the globally convergent ES's proposed in Chapter 4 already yielded encouraging results for unconstrained optimization, we have introduced a globalization procedure to include constraints. The latter ones are assumed in their most general form, meaning that the constraints can be relaxable and/or unrelaxable depending on the problem settings. The introduced procedure requires for relaxable constraints a merit function approach where we combined both the objective function and the constraints violation function. For the unrelaxable constraints two approaches were encompassed. In the first one, the objective function was evaluated directly at the generated sampled points, the feasibility was enforced using an extreme barrier function. The second approach projected the generated sampled points onto the feasible domain before evaluating the objective function. When the unrelaxable constraints are of the form of bounds on the variables or general linear inequality, we payed particular attention to the need to adapt or conform the generation of the ES offspring to the local geometry of the constraints, and tried to follow the globally convergent principles.

In the first part of our numerical experiments, we consider only unrelaxable constraints. We showed that our proposed ES approaches (using the extreme barrier or projection) can be competitive with state-of-the-art solvers for derivative-free bound and linearly constrained optimization. In the second part of our numerical experiments, we test our algorithms under the presence of both relaxable and unrelaxable constraints. On the chosen test problems, the merit approach showed promising results compared to the progressive barrier one [19], in particular, for relatively small feasible regions.

Chapter 6

Incorporating Local Models in a Globally Convergent ES

In this chapter, we show a possible way to incorporate quadratic surrogate models in our proposed ES to achieve a better performance. In general, various model techniques have been proposed to use with ES's. Jin [101] outlines a comprehensive survey of the most popular model-based techniques currently used with evolutionary algorithms, in particular, evolution strategies.

The modified ES algorithms, proposed in the previous chapters, evaluate the objective function at a significantly large number of points at each iteration, independently of its success or unsuccess. In a certain sense, they are even worse than opportunistic direct-search methods were polling is declared successful once a new better point is found (see Section 2.2). However, the previously evaluated points can be used in a number of ways to speed up the convergence and make ES type algorithms more efficient. The possibility that we will explore in this chapter is to use at the beginning of each iteration, a search step as in the search-poll framework of direct search [36]. For that purpose, a surrogate quadratic model of the objective function f can be minimized in a certain region using previously evaluated points. The surrogate models will be computed using techniques inspired from model-based methods for DFO (see Section 2.1). The latter methods have been shown to be more efficient and accurate than direct-search methods on chosen unconstrained test problems [125].

The proposed algorithm combines ES and model based techniques. Such a coupling has been first achieved for direct-search methods in SID-PSM [53] and then extended to MADS [48] with interesting results. The proposed hybrid algorithm has been designed to satisfy the convergence analysis of our globally convergent ES. We use quadratic interpolation to build our models. The minimization of the incorporated models is

expected to speed up the ES run. Our new hybrid algorithm follows the implementation choices suggested in [53].

This chapter is organized in the following way. In Section 6.1, we show how to incorporate local models into our proposed globally convergent ES, practical implementation issues are also emphasized. Section 6.2 reports our numerical experiments on both unconstrained and constrained optimization problems. Final conclusions are given in Section 6.3.

6.1 Incorporating local models in a globally convergent ES

In this work, we are interested in studying the impact of using quadratic models to enhance ESs. The new proposed algorithm can be described as follows: at the beginning of each iteration, a new search step will be taken where a quadratic model is minimized in a certain region. If the trial point y resulting from this process reduces sufficiently the objective function, meaning if $f(y) \leq f(x_k) - \rho(\sigma_k)$, then the search step and the current iteration are declared successful, the trial point is taken ($x_{k+1} = y$), the step size is left unchanged ($\sigma_{k+1} = \sigma_k$), and the ES main iteration step is skipped. If not, the iteration proceeds as in Algorithm 4.1. Similar to direct-search methods, the search step is optional and has no influence in the global convergence properties since (a) one can still easily prove that there are subsequences of unsuccessful iterations driving the step size to zero (refining subsequences), and (b) the analysis focuses then entirely on subsequences of unsuccessful iterations and those are only attainable by the ES mechanism itself (the poll step).

6.1.1 The general strategy of the search step

The search step is skipped if there are less than $n + 1$ previously evaluated points. We consider all the points previously evaluated and not only just those points that lead to a decrease in the objective function. Such choice has been shown to be a good strategy as it tries to capture as much information available as possible [53]. At the k-th iteration, given the current iterate x_k, the constructed quadratic model is minimized in a ball (or trust region) $B(x_k; \Delta_k) = \{x \in \mathbb{R}^n : \|x - x_k\| \leq \Delta_k\}$, centered at x_k with radius $\Delta_k = \theta\sigma_k$ (where θ takes the value 1 if the previous iteration was unsuccessful, or 2 otherwise). If no constraints are regarded, we use the standard Euclidean norm [49, 52], otherwise the infinity norm is used as a natural choice in the presence of bound constraints [49, 72]. The search step is enabled after a first quadratic model has been built, by minimizing the model in $B(x_k; \Delta_k)$. If no new model is formed at the current mean parent x_k,

then we use the last previously built model. The quadratic model can be built up to a maximum number of points of $(n+1)(n+2)$. If there are less points than needed for complete quadratic interpolation (meaning less that $(n+1)(n+2)/2$, see Section 2.1), one uses minimum Frobenius norm (MFN) models. MFN models consist on minimizing the Frobenius norm subject to the interpolation condition (see Section 2.1.2.3). When the number of points stored in the cache is in $((n+1)(n+2)/2, (n+1)(n+2)]$, two variants have been considered. In the first variant, only $(n+1)(n+2)/2$ evaluated points are used from the cache to compute a complete quadratic interpolation model by selecting the $(n+1)(n+2)/2$ nearest points. Such choice is shown to numerically perform better than if one selects 80% of the needed points as the nearest ones to the mean point x_k and the other 20% points are selected as the farther ones in an attempt to diversify the information used in the model computation [53]. In the second variant, we use all the evaluated points to build least-squares regression models.

6.1.2 Trust-region subproblem in the search step

The quadratic minimization problem corresponds to solve the following quadratic constrained optimization problem

$$\begin{array}{ll} \min\limits_{s\in\mathbb{R}^n} & m_k(x_k+s) \;=\; f(x_k) + g_k^\top s + \dfrac{1}{2} s^\top H_k s \\ \text{s.t.} & \|s\| \leq \Delta_k \end{array} . \tag{6.1}$$

Such problem is solved using the optimality conditions. Thanks to the quadratic form of the problem, one is able to characterize the global minimizer [49, 52]. Good algorithms exist for solving problem (6.1); such algorithms typically involve the computation of a full eigensystem and a Newton process applied to the secular equation [49]. The existing algorithms provide an accurate solution to the problem (6.1). However, they require several factorizations of H_k and thus for large-scale problems a different approach is needed. Meanwhile, such dimension constraint is out of the DFO context where one deals generally with comparatively small dimension problems. In our numerical experiments, we use the MATLAB routine `trust.m` to compute the solution of the optimization subproblem (6.1).

6.1.3 Geometry control in the search step

As emphasized in Section 2.1.2.2, one can use the condition number of the matrices $M(\bar{\phi}, \hat{Y})$ and $F(\bar{\phi}, \hat{Y})$ (depending on the number of the evaluated points) to monitor the poisedness (i.e, control the geometry of the sample set), where $\bar{\phi}$ is the natural basis of

monomials (2.6) and $\hat{Y}$ is a shifted and scaled version of the points set used Y such as $\hat{Y} \subset B(0;1)$.

In our setting, instead of controlling the condition number of $M(\bar{\phi}, \hat{Y})$ or $F(\bar{\phi}, \hat{Y})$, we used a singular value decomposition of the matrix, and replace all the singular values smaller than threshold ϵ by this threshold [53]. Such choice is motivated by the fact that (a) the search step is optional and used only to explore the local information independently of the quality of the constructed model, and (b) ignoring the geometry control of the points set may not deteriorate the performance compare to if one uses a geometry phase to monitor the poisedness of the points set [62].

6.1.4 Constraints treatment in the search step

The constraints are considered non-relaxable, meaning that $\Omega = \Omega_{nr}$ in the optimization problem 5.1. Their treatment in the search step can be done following two different approaches. The first one consists on using simply the barrier function f_Ω instead of f in the sufficient decrease condition to accept a new point y, meaning that the search step is declared successful only if $f_\Omega(y) \leq f(x_k) - \rho(\sigma_k)$. The second approach consists in projecting into the feasible domain Ω the outcome of the minimization subproblem (6.1) to yield the trial point y for the search step (and we will have $f_\Omega(y) = f(y)$). The projection approach is expected to lead to more progress than the barrier appeared, since all the successful search steps using the barrier variant are also successful for the projection one. However, the projection variant is known to be unpractical and expensive for general constraints [120], thus we propose to use the projection when it is doable, otherwise we switch to the barrier approach. For instance, in the case of bound constraints we will use the ℓ_2-projection (as it is trivial to evaluate), and in the case of general linear constraints we will use the ℓ_1-projection (as it reduces the projection to the solution of an LP).

6.1.5 Algorithm description

Algorithm 6.1 gives the complete description of the proposed strategy. The algorithm is split into two steps; a search step where one minimize a surrogate model, and a polling one where a main iteration of the globally convergent ES is performed. The poll step is launched only if the the search step has been unsuccessful. The step size parameter is updated only during the poll step, thus the convergence of the algorithm is exclusively controlled by the ES. In Chapter 4, three different ways to impose sufficient decrease conditions in ES are possible. We will adopt here only the mean/mean version that consists of applying sufficient decrease directly to the weighted mean x_{k+1}^{trial} of the new

parents, which has been shown to yield global convergence and to numerically perform the best among the different versions tested.

Algorithm 6.1: A globally convergent ES using a search step.

Initialization: Choose positive integers m_λ and m_μ such that $m_\lambda \geq m_\mu$. A starting point x_0. Choose initial step lengths $\sigma_0, \sigma_0^{\text{ES}} > 0$, an initial distribution $\mathcal{C}_0$, and initial weights $(\omega_0^1, \ldots, \omega_0^{m_\mu}) \in S$. Choose constants $\beta_1, \beta_2, d_{\min}, d_{\max}$ such that $0 < \beta_1 \leq \beta_2 < 1$ and $0 < d_{\min} < d_{\max}$. Select a forcing function $\rho(\cdot)$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. Search Step:

Try to compute a point with $f_\Omega(y) \leq f(x_k) - \rho(\sigma_k)$ by minimizing a surrogate model. If such point is found, then set $x_{k+1} = y$, declare the search step successful, and skip the poll step.

2. Poll Step:

2.1. Offspring generation: compute new sample points $Y_{k+1} = \{y_{k+1}^1, \ldots, y_{k+1}^\lambda\}$ such that

$$y_{k+1}^i = x_k + \sigma_k \tilde{d}_k^i, \; i = 1, \ldots, \lambda,$$

where the directions $\tilde{d}_k^i$'s are computed from the original ES directions d_k^i's (which in turn are drawn from a chosen ES distribution $\mathcal{C}_k$ and scaled if necessary to satisfy $d_{\min} \leq \|d_k^i\| \leq d_{\max}$).

2.2. Parent selection: evaluate $f_\Omega(y_{k+1}^i)$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\lambda\}$ by increasing order: $f_\Omega(\tilde{y}_{k+1}^1) \leq \cdots \leq f_\Omega(\tilde{y}_{k+1}^\lambda)$.

Select the new parents as the best μ offspring sample points $\{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\mu\}$, and compute their weighted mean

$$x_{k+1}^{trial} = \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i.$$

Evaluate $f(x_{k+1}^{trial})$.

2.3. Imposing sufficient decrease: if $f_\Omega(x_{k+1}^{trial}) \leq f_\Omega(x_k) - \rho(\sigma_k)$, then consider the iteration successful, set $x_{k+1} = x_{k+1}^{trial}$, and $\sigma_{k+1} \geq \sigma_k$ (for example $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\text{ES}}\}$).

Otherwise, consider the iteration unsuccessful, set $x_{k+1} = x_k$ and $\sigma_{k+1} = \bar{\beta}_k \sigma_k$, with $\bar{\beta}_k \in (\beta_1, \beta_2)$.

2.4. ES updates: update the ES step length σ_{k+1}^{ES}, the distribution $\mathcal{C}_k$, and the weights $(\omega_{k+1}^1, \ldots, \omega_{k+1}^{m_\mu}) \in S$. Increment k and return to Step 1.

6.2 Numerical experiments

6.2.1 Test strategy

We made a number of numerical experiments to measure the effect of incorporating local models in our proposed algorithms. We are mainly interested in observing the changes that occur in ES in terms of an efficient and robust search for stationarity.

The parameter choices are the same as in Section 5.3.1.2. We work with data profiles to assess the performance of the tested solvers using a maximal computational budget consisting of 1500 function evaluations as in [48, 53]. Data profiles are primarily interested in the behavior of the algorithms for problems where the evaluation of the objective function is expensive using data profiles. As for the levels of accuracy, we chose two values, $\alpha = 10^{-3}$ and $\alpha = 10^{-7}$.

6.2.2 Numerical results for unconstrained optimization

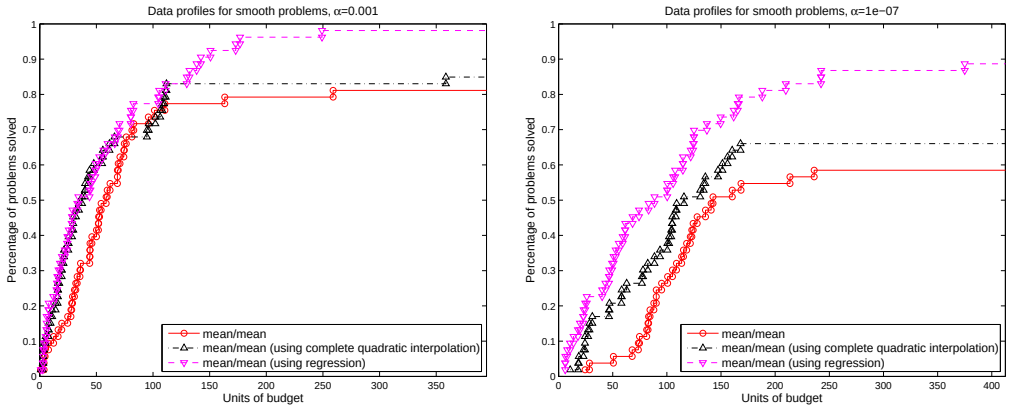
For our numerical experiments, we first compare the mean/mean version with and without the search step to confirm the expected advantage of incorporating local models. In a second part, we have compared the proposed algorithm and two other solvers BCDFO and SID-PSM. BCDFO [72] is a trust region interpolation-based code, this solver is shown to perform very well for both unconstrained and bound constrained optimization. SID-PSM [53, 54] is an implementation of a generalized pattern search method combined with quadratic polynomials to enhance the search step and with the use of simplex gradients to guide the function evaluations in the poll step. Our search step was mainly inspired by the SID-PSM implementation, thus a comparison with such solver seems natural to assess the impact of using local models.

Our test set $\mathcal{P}$ is the same as the one used in Chapter 4. The test problems have been considered in four different types, each having 53 instances: smooth (least squares problems obtained from applying the ℓ_2 norm to the vector functions); nonstochastic noisy (obtained by adding oscillatory noise to the smooth ones); piecewise smooth (as in the smooth case but using the ℓ_1 norm instead); stochastic noisy (obtained by adding random noise to the smooth ones).

6.2.2.1 Search step impact

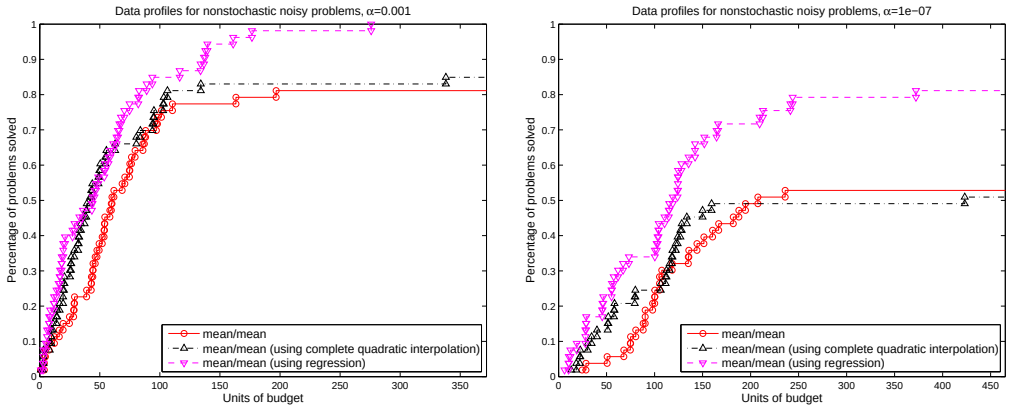
The purpose of this section is to quantify the impact of incorporating local model in the mean/mean version proposed in Chapter 4. As expected, our experiments have

shown that incorporating local models (i.e search step) improves the performance of our globally convergent ES.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.1: Data profiles computed for the set of smooth problems to assess the impact of incorporating local models, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

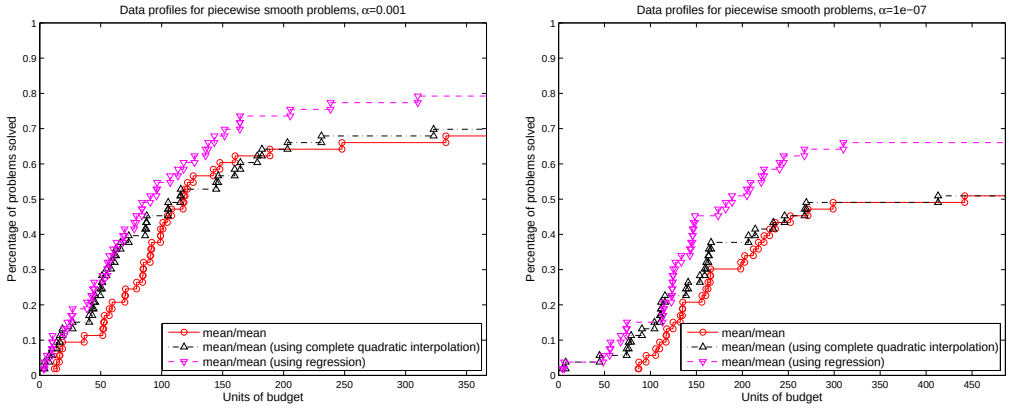


(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.2: Data profiles computed for the set of nonstochastic noisy problems to assess the impact of incorporating local models, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

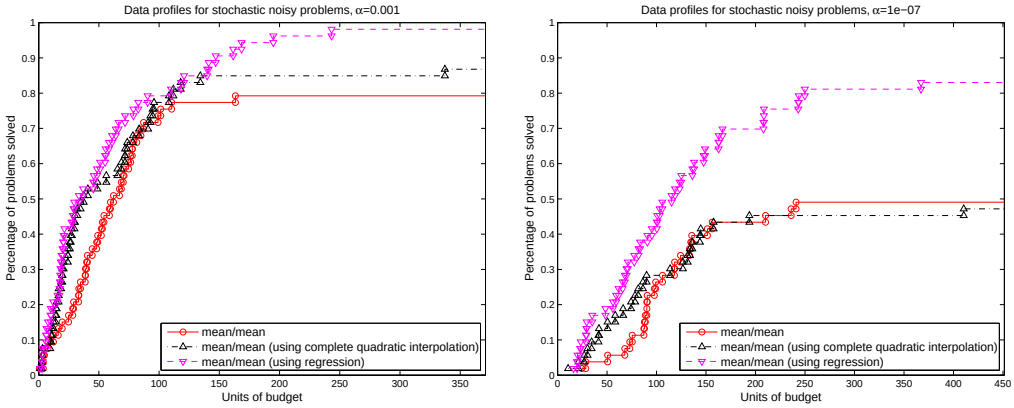
Figures 6.1, 6.2, 6.3, and 6.4 depict data profiles using two levels of accuracy 10^{-3} and 10^{-7}. The data profiles are clearly favorable to the incorporation of local models (i.e. mean/mean with a search step). Regression models are shown to improve significantly the performance of the mean/mean version for the tested problems. Complete quadratic interpolation models are not leading to significant improvement compare to the regression one, thus only regression models are going to be used further in the comparison with other solvers. For instance with an accuracy of 10^{-3} and when the problems are smooth (see Figure 6.1), the mean/mean version using regression models is able to solve about all the problems when the pure mean/mean version is solving around 80%. The

advantage of incorporating local models for higher accuracy, i.e. 10^{-7}, is more obvious. The results of the other class problems followed a very similar trend.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.3: Data profiles computed for the set of piecewise smooth problems to assess the impact of incorporating local models, considering the two levels of accuracy, 10^{-3} and 10^{-7}.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

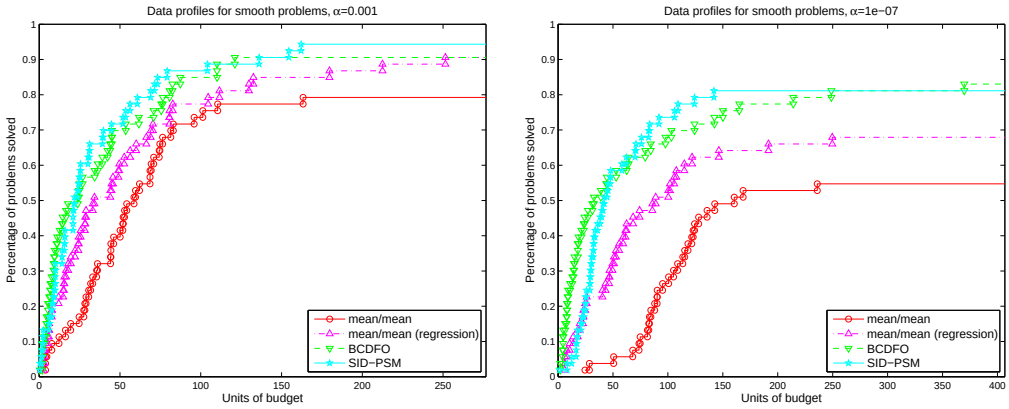
FIGURE 6.4: Data profiles computed for the set of stochastic noisy problems to assess the impact of incorporating local models, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

6.2.2.2 Comparison with other solvers

Figures 6.5, 6.6, 6.7, and 6.8 depict a comparison of our proposed algorithm with SID-PSM and BCDFO using two levels of accuracy 10^{-3} and 10^{-7}.

For smooth problems (see Figure 6.5), incorporating regression models leads to an improvement of the performance of the mean/mean version but not enough to outperform BCDFO and SID-PSM. For instance with an accuracy of 10^{-3} and for a unit budget of

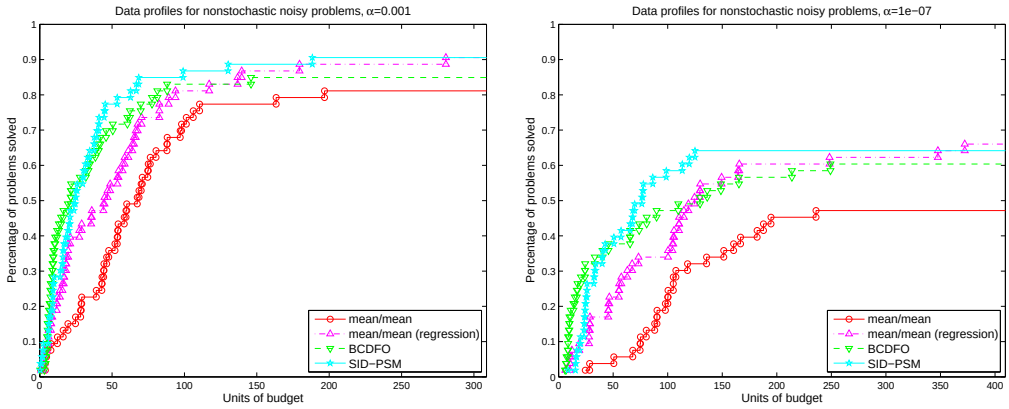
150 (i.e $150(n+1)$ function evaluations), BCDFO and SID-PSM are able to solve about 95% of the problems, the mean/mean version with regression models in the search step is solving around 85%. The pure mean/mean version is performing the worst by solving about 75% of the problems.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.5: Comparison with SID-PSM and BCDFO methods on the set of smooth problems using data profiles, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

When the optimization problems are noisy the mean/mean version (with regression models) shows better performance. For nonstochastic noisy problems (see Figure 6.6) the use of regression models leads the mean/mean version to perform better than BCDFO and have the same profile as SID-PSM for large budgets.

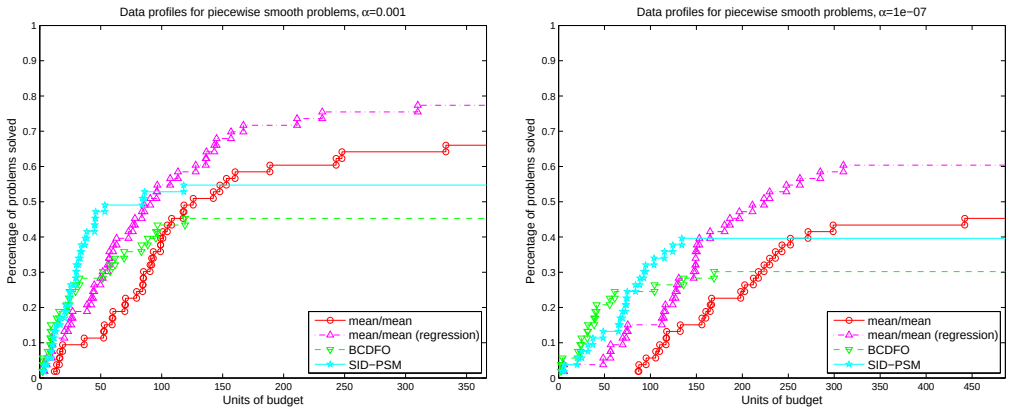


(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.6: Comparison with SID-PSM and BCDFO methods on the set of nonstochastic noisy problems using data profiles, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

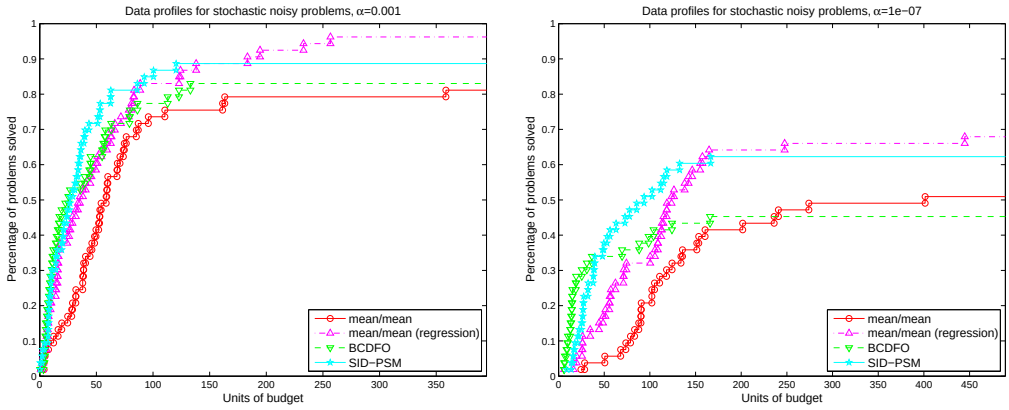
For piecewise smooth problem, see Figure 6.7, data profiles are clearly favorable to the mean/mean version with and without regression models in the search step. The

incorporation of such models leads to a very good performance. The leadership of the version mean/mean with regression models over all the tested solvers is confirmed for stochastic noisy optimization problems (see Figure 6.8).



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.7: Comparison with SID-PSM and BCDFO methods on the set of piecewise smooth problems using data profiles, considering the two levels of accuracy, 10^{-3} and 10^{-7}.



(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE 6.8: Comparison with SID-PSM and BCDFO methods on the set of stochastic noisy problems using data profiles, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

6.2.3 Numerical results for constrained optimization

For our numerical experiments only unrelaxable constraints are treated. In this section we are interested in measuring the impact of incorporating local models for constrained optimization problems. The local models are tested on ES-LC-B and ES-LC-P proposed in Chapter 5 to handle unrelaxable constraints. ES-LC-B is a solver handling general

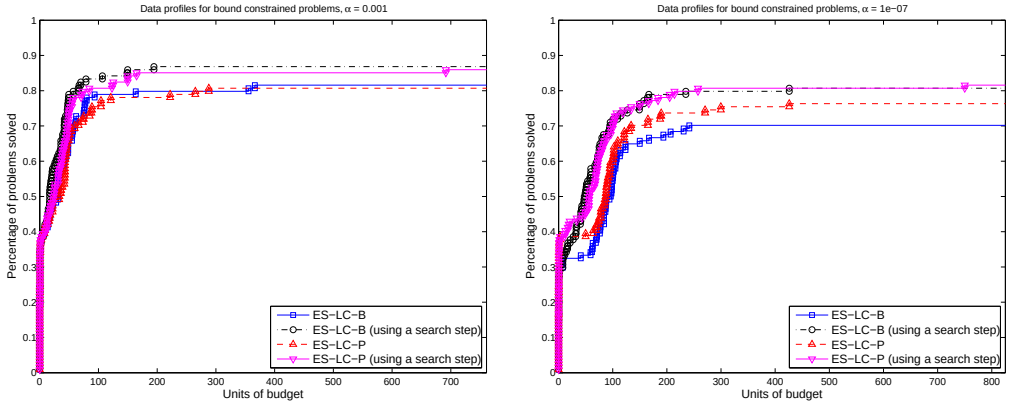
linear constraints essentially via a barrier function, the ES-LC-P solver is based on projection to enforce de feasibility.

We first compare ES-LC-B and ES-LC-P with and without the search step to confirm the expected advantage of incorporating local models. In a second part, we have compared the proposed algorithm and the same solvers used for the unrelaxable constraints comparison: PSWARM, MCS, CMA-ES, and BCDFO. The comparison with same solvers seems natural to assess the positive impact of using local models.

Our test set $\mathcal{P}$ in the same as the one used in Chapter 5 for unrelaxable constraints which is divided into two groups. The first group includes only pure bound constraints problems and it gathers 114 problems. The second group includes 107 general linear constrained problems.

6.2.3.1 Search step impact

The purpose of this section is to evaluate the impact of incorporating local model for constrained optimization problems. As expected, our experiments have shown that incorporating local models (i.e search step) improves the performance of our globally convergent ES, particularly for bound constrained optimization problems.

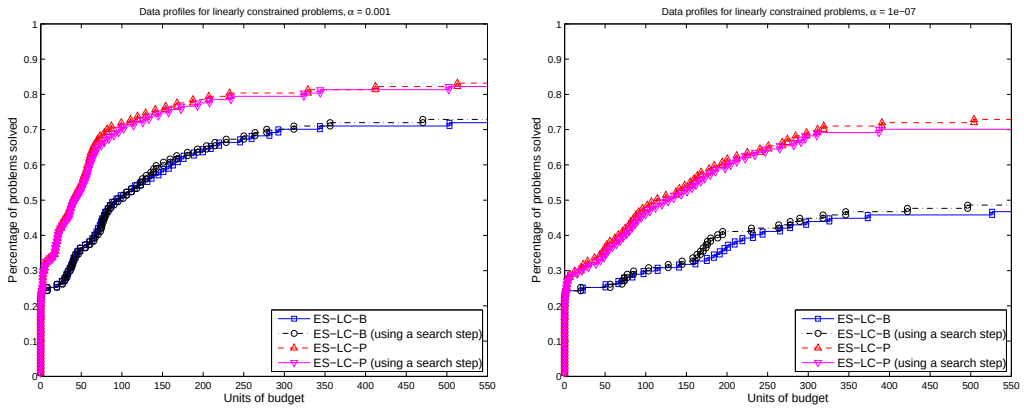


(a) Accuracy level of 10^{-3}.

(b) Accuracy level of 10^{-7}.

FIGURE 6.9: Data profiles computed for 114 bound constrained problems to assess the impact of incorporating local models, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

Figures 6.9 and 6.10 depict data profiles using two levels of accuracy for bound and linear constrained optimization problems. The data profiles are clearly favorable to the incorporation of local models (i.e. mean/mean with a search step), in particular for bound constraints. The improvement in the general linear constraints case is not significant on the tested problems. For instance with an accuracy of 10^{-3} when the



(a) Accuracy level of 10^{-3}.

(b) Accuracy level of 10^{-7}.

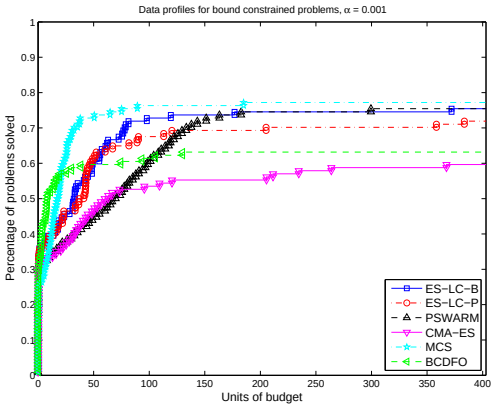
FIGURE 6.10: Data profiles computed for 107 general linearly constrained problems to assess the impact of incorporating local models, considering the two levels of accuracy, 10^{-3} and 10^{-7}.

problems are bound constrained (see Figure 6.9), regression models in the search step lead to an improvement of both solvers ES-LC-B and ES-LC-P. Both solvers using regression models are able to solve about 90% when the search step is disabled the same solvers are solving around 80%. The advantage of incorporating local models for higher accuracy, i.e. 10^{-7}, is more obvious. For general linear constraints (see Figure 6.10) the use of the search step does not lead to any significant performance improvement. Thus ES-LC-B and ES-LC-P are compared to other solvers only for bound constrained problems in the next subsection.

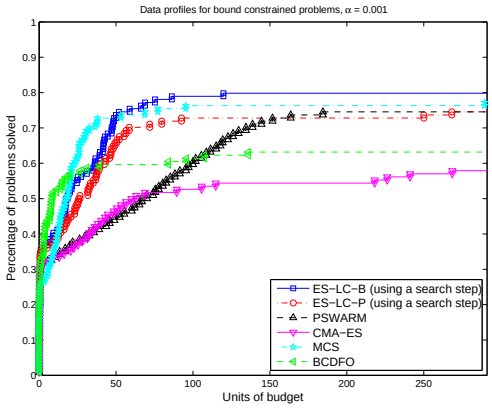
6.2.3.2 Comparison with other solvers

Figures 6.11 and 6.12 report a comparison of our solvers ES-LC-B and ES-LC-P with PSWARM, CMA-ES, MCS, and BCDFO using two levels of accuracy 10^{-3} and 10^{-7}. The purpose of this section is to outline to advantage of using local models to improve the numerical results emphasized for unrelaxable constraints in Section 5.3.1.

For an accuracy level of 10^{-3} (see Figure 6.11), incorporating models leads to an improvement of the performance of our solvers ES-LC-B and ES-LC-P. Thanks to the search step the ES-LC-B display the best performance of all the tested solvers for a sufficiently large budget. For instance for a unit budget of 150 (i.e $150(n+1)$ function evaluations), ES-LC-B is able to solve about 80% of the problems within a search step, MCS is solving around 75%. PSWARM and ES-LC-P are solving around 70% of the tested problems. CMA-ES and BCDFO are performing the worst by solving respectively about 50% and 60% of the problems. The same trend is followed for higher

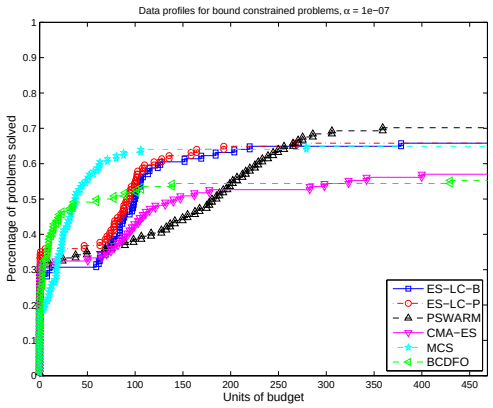


(a) The search step is disabled.

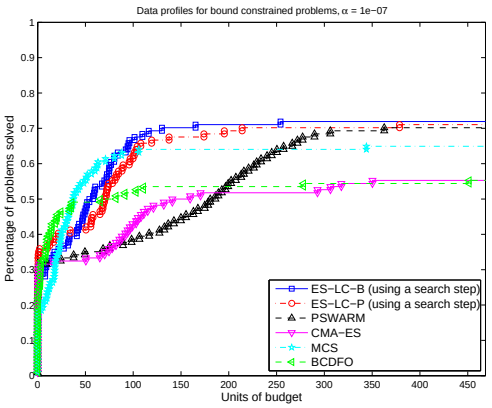


(b) The search step is enabled.

FIGURE 6.11: Data profiles for 114 bound constrained problems using an accuracy level of 10^{-3} (average objective function values for 10 runs).



(a) The search step is disabled.



(b) The search step is enabled.

FIGURE 6.12: Data profiles for 114 bound constrained problems using an accuracy level of 10^{-7} (average objective function values for 10 runs).

accuracy where the advantage of incorporating local models is more obvious for both solvers ES-LC-B and ES-LC-P(see Figure 6.12).

6.3 Conclusions

The main contribution of this chapter is to show that clear improvements due to the introduction of local models into our proposed ES have been demonstrated by numerical tests. First, on a set of unconstrained problems including smooth and noisy optimization problems, and then on a set of bound constrained problems for which our proposed globally convergent ES has shown very good performance and outperforms some of the state of the art algorithms in DFO area. The regarded approach for incorporating local

models consisted of using the points generated by the ES itself (during the poll steps) to build quadratic interpolation models. Regression local models were leading to better results compared to the complete quadratic interpolation models.

Chapter 7

Towards an Application in Seismic Imaging

Vibrations generated by earthquakes, explosions, or similar phenomena and propagated within the Earth or along its surface can yield information about the Earth and its subsurface structure. Such knowledge, called "Earth imaging", is of major interest for economy, environment and science. Geologists have developed several methods for Earth imaging using seismic wave information. Acoustic full waveform inversion is one of these procedures that attempts to derive high-resolution quantitative models of the subsurface using the full information of acoustic waves [167]. A complete description of the problem can be given as follows [161]. During the propagation, waves interfere with the environment and the total wavefield is recorded through a certain number of receivers (i.e. hydrophones or geophones). Since the waves are affected by the physical properties of the subsurface, they are carrying information about the environment that can be retrieved by an inversion process. The propagation waves are generated by sources situated in the domain of study (see Figure 7.1).

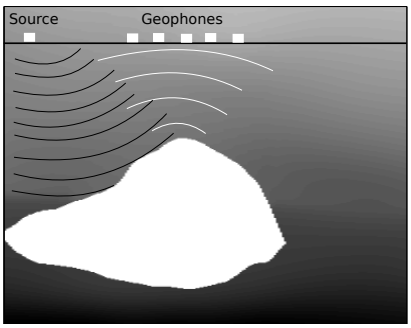


FIGURE 7.1: A graphical representation of acoustic waves propagated by a source are reflected by a reflective layer (in white) and are detected by the geophones. The reflective layer in this example represents a salt dome.

Acoustic full waveform inversion has been almost exclusively used by academic researchers but it is now adopted by practitioners. Nevertheless, the computational cost is still expensive compared to other methods employed for seismic exploration. The attraction of the approach is the promise of deriving high-fidelity earth models for seismic imaging from the recoder full waveforms. As our ability to understand and manage the complex, non-linear inversions has been developed and the computer power available has grown, full waveform inversion has become more and more practical.

Acoustic full waveform inversion (FWI), formulated as a non-linear optimization least-squares minimization, can be efficient when the starting propagation velocity model is accurate enough, but otherwise suffers from stalled convergence to spurious local minima due to the oscillatory nature of the data [126]. The main step of FWI in seismic imaging is to find a good starting point without the need for sophisticated a priori knowledge on the velocity model. A good starting point has to explain the data within a low frequency, meaning that a smooth version of the true velocity model can be seen as a good starting velocity model [38, 167]. First-arrival travel-time tomography [131], stereotomography [114] or recently Laplace domain inversion [152] are typically the most used methods to generate an accurate initial propagation velocity model. Multi-scale strategies are also used to mitigate the non-linearity and reduce the dependence on the starting velocity model of FWI [139, 156].

In this thesis, we propose another alternative to find an initial velocity model for FWI without any physical knowledge. Motivated by the recent growth of high performance computing (HPC), we will tackle the high non-linearity of the problem to minimize and find a good starting velocity model, using evolution strategies (ES's) that are known to be easy to parallelize. In general, global optimization methods have been already used to solve such a problem. A first attempt was through simulated annealing to invert the ocean bottom properties [47]. A second application came from Gerstoft [70], who used genetic algorithms to invert seismoacoustic data. The main drawbacks of global optimization methods were that they are very consuming in terms of the objective function evaluations and very depending on the parametrization of the methods. In the prespecified examples (genetic algorithms and simulated annealing), the methods were based on a fast objective function evaluation as well as a very simple parametrization of the model. Unlike the previous works, the objective function in this chapter is regarded as a black box hiding the problem complexity.

The first contribution of this chapter is to adapt ES's to the FWI setting. The velocity model is represented as faithfully as possible, while limiting the number of parameters needed, since each additional parameter is an additional dimension to explore. The

second contribution is to propose a highly parallel ES adapted to the FWI setting. The initial results, obtained in this direction, are emphasized in the numerical section.

7.1 Full-waveform inversion

Estimating the subsurface velocities from a seismic recording is the main aim of FWI. One uses the recorded wavefield to guess the physical properties of the medium which the wavefield have propagated through. The wave propagation depends on the medium properties inside a bounded parallelepiped domain $\Omega \subset \mathbb{R}^3$. Two main approaches are traditionally used for finding the solution for FWI, (a) to consider the problem in the time-domain or (b) in the frequency-domain. Details on both approaches can be found in [38, 167], the frequency-domain approach is more advantageous when solving the full-waveform inversion, in this thesis we address this approach only [167].

7.1.1 Forward problem

Given the medium properties (e.g. the subsurface velocity), the forward problem consists of modeling the full seismic wavefield at any time and location. The wave propagation is controlled by a partial differential equation. The formulation of the equation depends on the characteristics of the propagation model. In the acoustic case, the seismic wavefield $u(x)$ at the position $x \in \Omega$ in the frequency-domain is governed by the so-called heterogeneous Helmholtz equation defined by:

$$-\Delta u(x) - k^2(x)u(x) = s(x), \tag{7.1}$$

where $k(x) = 2\pi f/m(x)$ is known as the wavenumber, $f \in \mathbb{R}$ is the regarded frequency, and $m(x)$ is the propagation velocity model. Δ denotes the Laplacian operator, and $s(x)$ is a source term.

The most popular direct method to discretize Equation 7.1, is the finite-difference method, but finite-element or finite-volume approaches can be used too [38]. In this thesis, we use a uniform second-order accurate finite-difference technique with 7 points in the Cartesian three-dimensional grid, which is known to be cheap to compute [133, 135]. Since the domain Ω is supposed to be bounded, one must add an absorbing boundary in order to simulate properly the wave propagation phenomena. Perfectly Matched Layers (PML) [29] is one of the most used boundary treatment. The wave equation can be then reduced to a linear system of the form:

$$Au = s, \tag{7.2}$$

where $u, s \in \mathbb{C}^N$ represent respectively the vectorization of the seismic wavefield and the source term using a lexicographical ordering. N is the dimension of the problem (i.e. the number of points in the regarded domain Ω after discretization). $A \in \mathbb{C}^{N\times N}$ (the impedance matrix) is a sparse matrix (with only seven diagonals of non-zeros) and is neither symmetric nor Hermitian due to the absorbing boundary conditions [133]. The matrix A embeds the properties of the subsurface and depends on the propagation velocity model m that we want to quantify. In the case of multi-sources (e.g. p source terms) and for a given frequency f, we obtain a block linear system to solve of the form:

$$AU = S, \tag{7.3}$$

where $U, S \in \mathbb{C}^{N\times p}$. In this thesis, we will not consider multiple frequencies simultaneously since the discritization step depends on the chosen frequency via a stability condition [46] in the following way:

$$h \leq \frac{m(x)}{n_\lambda f} \quad \forall x \in \Omega,$$

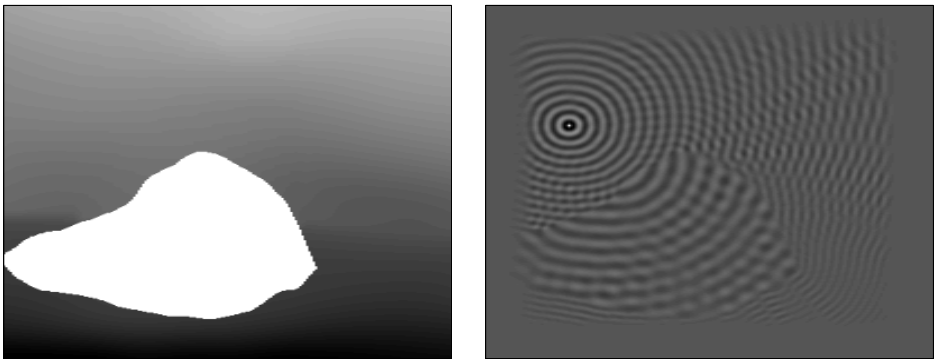
where n_λ is the number of points per wavelength. Such number is generally chosen sufficiently large to avoid dispersion errors in the solution. For instance, considering a second-order 7 points discretization scheme, one chooses $n_\lambda \in [\![10, 12]\!]$. Using a higher order discretization scheme, it is shown that one can use smaller n_λ without a significant dispersion error in the solution [133] (e.g. $n_\lambda = 4$). The smaller n_λ is, the larger the discretization step h is, and the smaller is the problem size to solve.

Two main approaches are traditionally used for solving the linear system (7.3): direct or iterative solvers. Direct solvers operate through a decomposition of the matrix A as the product of a lower triangular matrix and an upper triangular matrix (LU). The problem (7.3) can be then solved directly by forward and backward substitutions applied separately to the source term. Direct solvers are known to be efficient for multiple sources as the decomposition is performed only once for all the term sources. However, the LU decomposition becomes very expensive in time and memory for large scale problems. Such constraints prevent the use of direct solvers for large scale problems (e.g. a 3D wave propagation problem at high frequency range) [133]. The other alternative is guaranteed through iterative solvers. The latter ones are implemented generally using preconditioned Krylov subspace methods [148]. The main advantage of the iterative solvers is the low memory requirement, their main drawback results on the difficulty to find an efficient preconditioner.

In this thesis, we use a direct solver at low frequencies ($f \leq 2\,\text{Hz}$) and an iterative solver as far as the frequency range gets higher ($f > 2\,\text{Hz}$). As a direct solver we use MUMPS

(MUltifrontal Massively Parallel Sparse direct solver) [9, 10] which implements a direct method based on a multi-frontal approach by performing an LU decomposition. For the iterative solver, we use DMBR (Deflated Minimal Block Residual) [41, 113] a variant of BGMRES (Block Generalized Minimal RESidual) [146] in which a deflation technique is used to discard subspaces at the beginning of each iteration. As a preconditioner we use a geometric two-level preconditioner adapted to the specificity of our problem [40, 135].

Figure 7.2 outlines a graphical representation of a 2D forward problem solution over a slice of the velocity model. The wave propagation is affected by the velocity model properties, it is carrying information about the subsurface : we remark in particular the existence of a reflected layer which can correspond to a salt dome zone. On the border of the domain Ω, one can observe properly the absorbing boundary condition on the waves propagation using the PML. The interference of the waves with the reflected layer will generate reflection waves. The later are recorded at different time steps using geophones to generate the so-called *seismograms*, meaning the observed data for the associated inverse problem (see Section 7.1.2).



(a) A 2D velocity model. (b) Wave propagation.

FIGURE 7.2: A graphical representation of acoustic wave propagation over a two-dimensional velocity model.

7.1.2 FWI as a least-squares local optimization

In the frequency domain, FWI in its standard form (for a fixed frequency value f) tries to minimize the least-squares misfit [161]:

$$\mathcal{C}(m) = \frac{1}{2}\sum_{i=1}^{p} (\Delta d^i)^\dagger W^i \Delta d^i, \tag{7.4}$$

where $\dagger$ denotes the adjoint operator (transpose conjugate), and p is the number of source terms. The weight matrices W^i are in general used to include a priori data

information. The misfit vector $\Delta d^i(m) = d^i(m) - d^i_{\text{obs}}$ for the i-th source of dimension n is computed as the difference at the receiver positions between the recorded seismic data d^i_{obs} (i.e. seismograms) and the modeled seismic one $d^i(m)$. The latter is related to the modeled seismic wavefield u^i (computed as the i-th column of the U solution of (7.3)) projected using the operator $\mathcal{P}_{\text{data}}$, which extracts the values of the wavefield at the receiver positions for each source; $d^i(m) = \mathcal{P}_{\text{data}}(u^i)$. The projection operator makes the FWI an ill-posed problem, meaning that an infinite different number of velocity models matches the data, leading to the same objective function value. Therefore, an additional regularization term is classically added to the inversion problem to make it well posed [161]. In addition to the velocity model, the source excitation is generally unknown and must be included as an unknown of the problem [140].

Around a starting velocity model m_0, FWI minimization is solved by perturbing m_0 with a perturbation model Δm. Using a second-order Taylor-Lagrange expansion of the misfit function $\mathcal{C}$ around m_0, the minimum of the misfit near to m_0 is given by the following perturbation velocity model vector [167] (Newton update):

$$\Delta m = -\left[\frac{\partial^2 \mathcal{C}(m_0)}{\partial m^2}\right]^{-1} \frac{\partial \mathcal{C}(m_0)}{\partial m}, \tag{7.5}$$

for the expression of $\frac{\partial^2 \mathcal{C}(m_0)}{\partial m^2}$ and $\frac{\partial \mathcal{C}(m_0)}{\partial m}$ the reader is referred to the normal equations in [38, 167] and the references therein. Note that FWI is a non-linear optimization problem, thus using the velocity update (7.5) one needs to iterate more than once until a stopping convergence criteria is reached.

The term $\frac{\partial^2 \mathcal{C}(m_0)}{\partial m^2}$ is very expensive to compute, thus alternatively in practice the inverse of the Hessian in Equation 7.5 is replaced by a scalar α (i.e. a step size) leading to the steepest-descent method. The step size is then estimated by backtracking a line search along the steepest descent direction given by the gradient of the objective function $\frac{\partial \mathcal{C}(m_0)}{\partial m}$ [68, 161]. Using an adjoint-state method [130] applied to a Lagrangian function corresponding to the misfit function $\mathcal{C}$ augmented with the forward problem (Equation 7.3), the expression of the misfit function gradient at the point m_0 can be deduced in the following way [136]:

$$\frac{\partial \mathcal{C}(m_0)}{\partial m} = \sum_{i=1}^{p} \mathcal{R}\left[(u^i)^\top \left[\frac{\partial A}{\partial m}\right]^\top A^{-1} \tilde{\mathcal{P}}_{\text{data}} W^i \overline{\Delta d^i}\right], \tag{7.6}$$

where $\tilde{\mathcal{P}}_{\text{data}}$ is the operator that projects $\overline{\Delta d^i}$ onto the forward problem space and $\mathcal{R}(\cdot)$ is the real part of the referred vector. The term $A^{-1}\tilde{\mathcal{P}}_{\text{data}} W^i \overline{\Delta d^i}$ corresponds to the so-called *backward problem*, where we solve the same linear system as in the forward problem (7.2) except that the source term s is replaced by $\tilde{\mathcal{P}}_{\text{data}} W^i \overline{\Delta d^i}$.

Algorithm 7.1: A multi-scale algorithm for frequency-domain FWI.

Initialization: Let an initial velocity model m_0 and choose an initial step size $\alpha_0 > 0$. Set $k = 0$.

for frequency = f low to f high **do**

Until some stopping criterion is satisfied:

1. Solve the forward problem.
2. Solve the backward problem.
3. Compute the gradient of the objective function.
4. Estimate the step size α_k.
5. Update the velocity model : $m_{k+1} = m_k + \alpha_k \Delta m_k$.

end for

The non-linearity and the ill-posedness of the FWI problem are in practice tackled in the frequency domain using a multi-scale approach where one starts the inversion within a low frequency range to mitigate the non-linearity of the inversion [155], and then incorporate progressively higher frequencies in the inversion process (see Algorithm 7.1). The frequency range used in FWI is from 1 Hz to 15 Hz.

7.2 ES for building an initial velocity model for FWI

7.2.1 Methodology

FWI as presented – a minimization of a least-squares local optimization problem – crucially depends on the starting velocity model m_0. In fact, the FWI is converging to satisfactory results only when the starting velocity model is situated not far from the global minimum [167]. Before applying FWI, a starting model is generally built. The most used techniques are first-arrival travel-time tomography (FATT) [131], stereotomography [114] or recently Laplace domain inversion [152]. FATT is a method that for many years has proven to be stable in generating smooth velocity models of the subsurface. Some examples of application of FWI to real data using a starting model built by FATT are shown in [134, 141]. The stereotomography is regarded as one of the most promising methods for building a smooth velocity model. It exploits the arrival time of locally coherent events within an automatic procedures to select a seismogram collection [114]. Some applications to synthetic and real data sets are shown in [34, 35]. The Laplace domain inversion can be viewed as a frequency domain inversion using a

complex valued frequency where the real part is chosen as zero and imaginary part controls the time damping of the seismic wavefield. Applications of Laplace domain FWI to synthetic and real data are shown in [152–154].

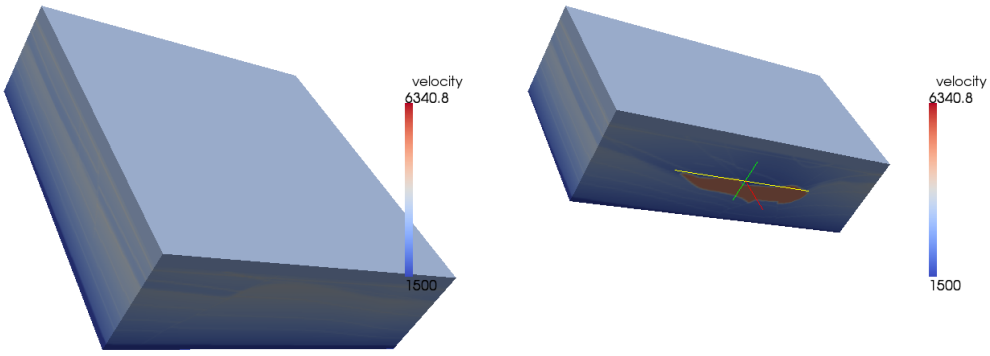
Motivated by the recent growth of high performance computing (HPC), we will try to find a good starting velocity model using ES's that are known to be naturally parallelizable. In our context, we work with CMA-ES, see Section 3.2.3, which is regarded as state of the art algorithm for numerical blackbox optimization if we assume that a sufficient budget is available (large number of objective function evaluations). The CMA-ES algorithm has shown superior performance on difficult ill-conditioned, non-separable and highly multi-modal problems [23, 145]. The main drawback of CMA-ES is its need for a large budget to give outstanding results. The modifications we proposed in the previous chapters lead to a significant reduction on the convergence cost (i.e. meaning the number of objective function evaluations needed to converge to a stationary point). Thus in our proposed implementation, CMA-ES is hybrid with our proposed modifications to speed up the algorithm.

ES's can have great success on problems that are known to be computationally difficult, good results have been found in terms of the quality of the minimum found. However, most types of ES's suffer from the *curse of dimensionality*, meaning that their performance is good on low dimensional problems, but deteriorates as the dimensionality of the search space increases [127]. For realistic simulations of FWI, the size of the velocity models N in general exceeds 10^6, thus trying to solve directly the problem using CMA-ES is out of the scope of the method. However, our purpose is not to solve FWI but only to find a good starting velocity model which can be later improved using FWI procedure. The initial velocity model m_0 is needed just to represent the general structure of the true model, such representation is generally smooth and can be expressed using only few parameters [167]. Once we find an efficient procedure to represent the velocity model using minimum model parameters, the ES method will try to find the parameters that lead to a smooth representation of the velocity model we are trying to invert.

7.2.2 SEG/EAGE salt dome velocity model

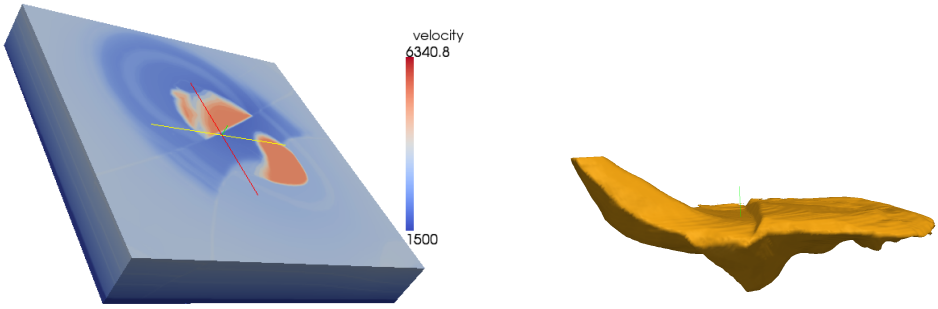
In this thesis, all our numerical experiments are performed using a 3D academic example of a velocity model, known in the geophysics community by SEG[1]/EAGE[2] salt dome velocity model outlined in Figure 7.3. The velocity model is based on a typical US Golf coast salt structure. Special care was taken to ensure that the model is geologically feasible and would be an adequate testing mechanism for seismic imaging algorithms.

[1]The Society of Exploration Geophysicists.
[2]European Association of Geoscientists and Engineers.



(a) The full velocity model.

(b) A vertical section.



(c) An horizontal section.

(d) The salt dome.

FIGURE 7.3: Academic 3D SEG/EAGE salt dome velocity model using Paraview [88]. The geophysical domain size is of $20 \times 20 \times 5\,\mathrm{km}^3$ in which the minimal velocity is of 1500 m/s. The velocity model is representing a dome of salt in the subsurface of Earth, which abruptly increases the velocity of propagation of the compressional waves.

The seismic waves propagate in water and salt dome of the model at the minimal and maximal velocities of 1500 m/s and 4418 m/s, respectively.

The geological domain size is of $20 \times 20 \times 5\,\mathrm{km}^3$, then if $n_\lambda = 10$ (i.e. using 7 points discretization scheme for the forward problem) and with a discretization step $h = \frac{1500}{10f}$ (which respects the stability condition (7.4)), the size N of the forward problem will be of $136f * 136f * 34f$ where f is the working frequency (i.e. $N \approx 10^6 * f^3$).

On the SEG/EAGE salt dome velocity model, we will test our parametrization procedure to deduce the appropriate basis leading to an accurate smooth representation using a minimum number of parameters. On the reduced search space, the implemented ES will try to find a smooth velocity model version of the SEG/EAGE salt dome model.

7.2.3 Search space reduction

To reduce the search space we will try to investigate how to represent a realistic propagation velocity model using a minimum of parameters. The search space reduction has been investigated over the past using subspace approaches [104, 132, 157]. In a subspace approach, one basically tries to restrict the search space for only some specific directions. In FWI context, the velocity model perturbation $\Delta m \in \mathbb{R}^N$ is restricted to lie in an n-dimensional subspace of $\mathbb{R}^N$ which is spanned by the vectors $\{v_i\}_{i=1,\dots,n}$ where $n << N$. The model perturbation can then be written as follows:

$$\Delta m = \sum_{i=1}^{n} y_i v_i = V y, \tag{7.7}$$

where $y \in \mathbb{R}^n$ are the new parameters to invert, and $V = [v_1, \dots, v_n] \in \mathbb{R}^{N \times n}$ is the so-called *reduction basis*. Subspace approaches lead to an important simplification of the problem [104], but they are very sensitive to the choice of the reduction basis. In fact, by restricting the search space to specific directions, the neglected ones may be the vectors which are important in finding the global minimum of the objective function $\mathcal{C}$ (see Equation 7.4). Often researchers use sinusoidal basis as reduction basis and try to find a vector parameter y which produces acceptable agreement to the observation [132].

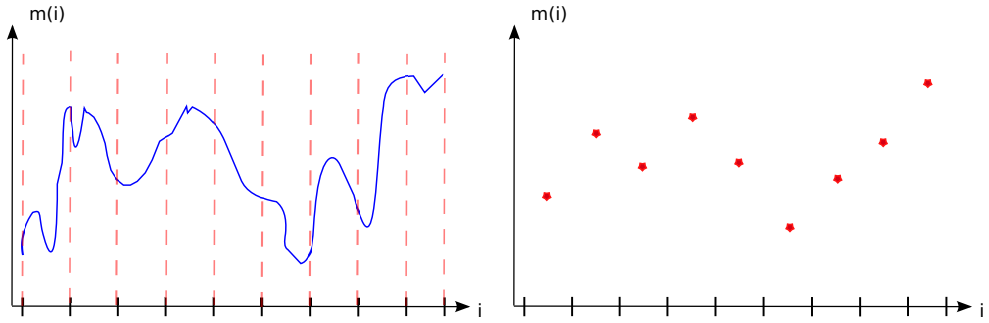
Inspired by the technics used in image compressing, we propose in this thesis a new procedure to construct this basis. We propose to use a sinusoidal and rectangular basis functions, more specifically Discrete Cosine Transform (DCT) [6] and Haar wavelets [55]. The Haar wavelet transform is used to magnify the vector parameter $y \in \mathbb{R}^n$ to fit the original space $\mathbb{R}^N$, but it leads to a pixelization effect. The DCT is applied to produce a smooth velocity model and reduce the pixelization effect introduced by the Haar wavelets. For a reduced 3D velocity model of size $n = n_x \times n_y \times n_z$, the magnification procedure will be adapted to the 3D geometry of the velocity model. For ease of exposition, we will first explain our chosen approach for the one-dimensional case, and then give a generalization to cover the realistic 3D geometry.

7.2.3.1 One-dimensional approximation procedure

We suppose that one disposes of a vector $m \in \mathbb{R}^N$ which we are willing to represent by $y \in \mathbb{R}^n$ with $n << N$ (i.e. reduction). Being able to construct the model parameters m using only the y parameters is also our target (i.e. magnification). Actually for the inverse problem, the reduction will not be used (i.e. meaning compute n from N parameters) since one has no a priori knowledge on the real velocity model m, only the parameters in $y \in \mathbb{R}^n$ are used to build the velocity vector m of size N. In our context,

the reduction operation will be only used to estimate how efficient is our magnification procedure.

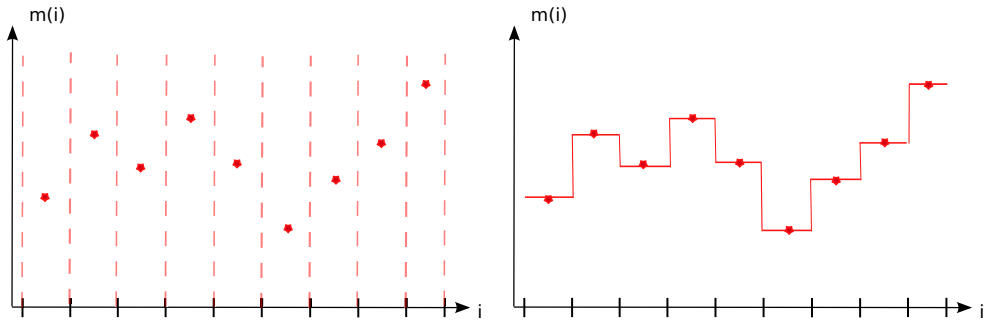
The reduction procedure is inspired from the Haar wavelets procedure [55], meaning that from a vector represented by $N = 2^q$ parameters with $q \in \mathbb{N}$, we consider simply to pair up the parameters and replace each pair by the average of the two parameters in each pair. The computed vector will be of half size compared to the original vector and will contain only the pair average of the initial vector. The procedure is repeated until we get a number of n parameters. The Haar procedure assumes that both the numbers of parameter N and n are of the form 2^q for some $q \in \mathbb{N}$, such assumption in image processing context is not an obstacle as most images are a power of two. In seismic imaging context, instead of an ordinary image one works with a velocity model for which the number of parameters is far to be a power of two. To overcome such problem and given a vector $m \in \mathbb{R}^N$, we simply propose to decompose the velocity vector $m \in \mathbb{R}$ to n subdivisions (see Figure 7.4(a)). Each subdivision will be represented by one value computed as the average of all the parameters included in this subdivision (see Figure 7.4(b)).



(a) The subdivision procedure. (b) The reduced velocity vector.

FIGURE 7.4: The reduction procedure over a one-dimensional case.

The duplication procedure consists in building a vector m of size N using a small-size vector $y \in \mathbb{R}^n$ with $n << N$. For this sake, we construct first an empty vector m of size N with n subdivisions. Each subdivision contains around $\delta = [\frac{N}{n}]$ parameters. The n parameters of the velocity vector y are then distributed over the n subdivisions (see Figure 7.5). The value associated to each subdivision is then duplicated all over δ parameters assigned for that subdivision. The duplication procedure, as presented, introduces a pixelization effect over the constructed vector m (see Figure 7.5(b)). A possible improvement on the quality of duplication can be made through a DCT transform to improve the approximation procedure and omit the subdivision discontinuities.



(a) The subdivision procedure. (b) The duplicated velocity vector.

FIGURE 7.5: The duplication procedure over a one-dimensional case.

The magnification procedure aims in general at removing noise or producing a less pixelated image. The most used smoothing algorithms are Gaussian smoothing [5], bilateral filters [162] and sinusoidal based approaches [6]. As a smoothing procedure, we choose to work with sinusoidal basis as one of the most techniques used on the subspace approaches for FWI to generate a smooth approximation vector using few coefficients [132]. We will smooth the pixelization effect in the magnified vector (see Figure 7.5(b)) using a discrete cosine transform (DCT) [6]. Assuming that we have a vector y of size n, we consider an n subdivision $[x_i, x_{i+1}]$ of indices, see Figure 7.6.

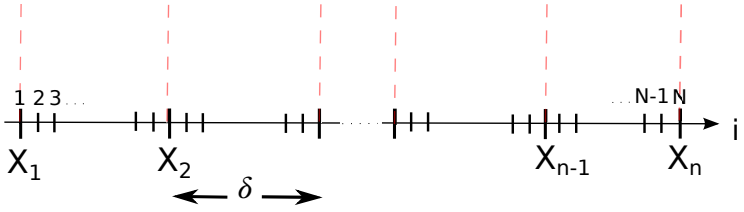


FIGURE 7.6: An illustration for index subdivisions.

Inside each subdivision $[x_i, x_{i+1}]$ all the points have the same value as $y(i)$ (see Figure 7.5(b)). Such requirement will be imposed on the magnified velocity vector m as if one has the same mean value as the original vector y, which can be explicitly expressed by [121]:

$$\frac{1}{x_i - x_{i+1}} \int_{x_{i+1}}^{x_i} m(x)dx = y(i) \quad i = 1, \ldots, n. \tag{7.8}$$

The vector $m \in \mathbb{R}^N$ is expressed using a discrete cosine basis of $\mathbb{R}^n$ in the following way:

$$m(x) \quad = \quad \sum_{j=1}^{n} a_j \cos\left(\frac{(j-1)\pi}{N}(x-1)\right), \tag{7.9}$$

where $a = (a_j)_{1 \leq j \leq n} \in \mathbb{R}^n$. All the subdivisions are supposed to have the same length $\delta = [\frac{N}{n}]$, thus $\delta = x_{i+1} - x_i$ and $x_i = (i-1)\delta + 1$. By incorporating equation (7.9) in

the condition (7.8), we obtain the vector a by solving a linear system of the form

$$Ca \quad = \quad y, \tag{7.10}$$

where $C \in \mathbb{R}^{n\times n}$ is a matrix such as

$$C(i,j) = \begin{cases} 1 & \text{if} \quad j = 1, \\ \frac{2N}{(j-1)\pi\delta} \cos\left(\frac{\pi}{N}(j-1)(i-\frac{1}{2})\delta\right) \sin\left(\frac{\delta\pi}{2N}(j-1)\right) & \text{otherwise.} \end{cases}$$

The coefficient matrix C is of a small size and nonsingular [121], thus the inversion cost is negligible. The one-dimensional smoothed vector m is then built by evaluating (7.9) for all $i \in \{1, \ldots, N\}$:

$$m(i) \quad = \quad \sum_{j=1}^{n} a_j \cos\left(\frac{(j-1)(i-1)\pi}{N}\right),$$

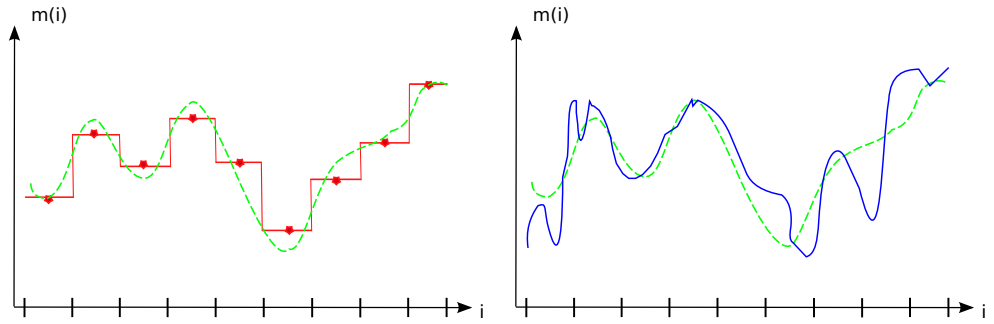
or equivalently

$$m \quad = \quad My, \tag{7.11}$$

where $M = KC^{-1} \in \mathbb{R}^{N\times n}$ and $K \in \mathbb{R}^{N\times n}$ is a matrix defined such as $K(i,j) = \cos(\frac{(j-1)(i-1)\pi}{N})$. The vector $y \in \mathbb{R}^n$ is the original vector before magnification (as the magnification procedure leads to the vector $m \in \mathbb{R}^N$). Equation 7.11 shows that the magnification procedure corresponds to a linear operator. Figure 7.7 outlines an illustration of the proposed algorithm applied to a one-dimensional vector. Compared to the duplicated velocity vector using the Haar transform, the smoothing effect of DCT transform on improving the quality of approximation is clear and leads to a better representation of the true velocity vector (see Figure 7.7(b)).

7.2.3.2 Three-dimensional approximation procedure

A multidimensional transform can be basically ensured using a composition of the one-dimensional magnification procedure along each dimension [160]. Equation (7.11) can be immediately extended to two-dimensional or three-dimensional velocity model. A detailed description of the extension of Equation (7.11) to higher dimensions is given in [160]. In the case of three-dimensional data, suppose that we have a small 3D velocity model y of $n = n_x \times n_y \times n_z$ parameters, we ought to build a magnified 3D velocity model m of size $N = N_x \times N_y \times N_z >> n$ parameters. The magnification procedure is obtained by applying Equation (7.11) consecutively to first the x axis, then y, and



(a) Graphical comparison between the duplication (solid line) and magnification (dashed line) procedures.

(b) A comparison between the magnified velocity vector (dashed line) and the true velocity vector (solid line).

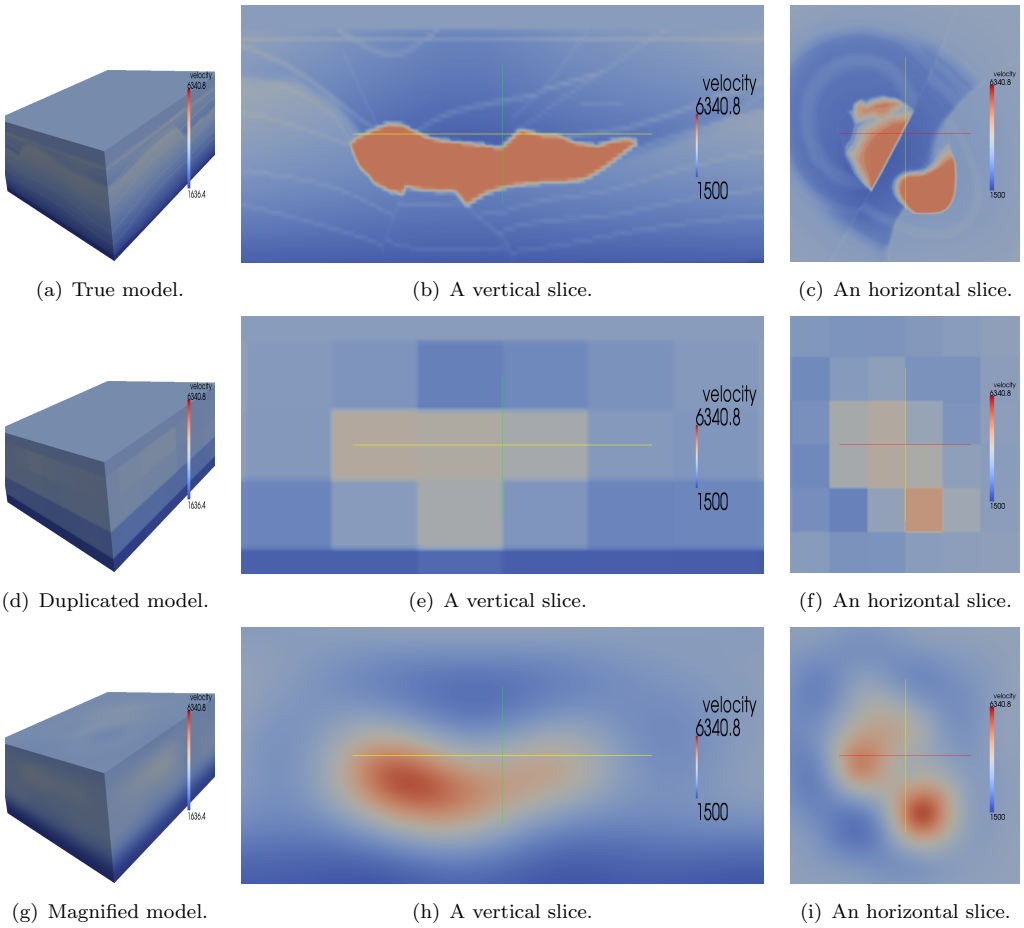
FIGURE 7.7: A one-dimensional magnification procedure using DCT transform. Compared to the duplicated vector, the magnification using DCT transform represents better the true velocity vector.

finally z as follows:

$$\begin{aligned} T(:,:,k) &= M_x[y(:,:,k)]M_y^\top \quad k = 1,\ldots,n_z \\ m(i,:,:) &= T(i,:,:)M_z^\top \quad i = 1,\ldots,N_x \end{aligned}$$

where $M_x \in \mathbb{R}^{N_x \times n_x}$, $M_y \in \mathbb{R}^{N_y \times n_y}$, and $M_z \in \mathbb{R}^{N_z \times n_z}$ are the one-dimensional smoothing matrices defined in Equation (7.11) along the axes x, y, and z, respectively.

To illustrate numerically the performance of the three-dimensional approximation procedure (i.e. smoothing and magnification), we used the SEG/EAGE salt dome velocity model (see Figure 7.3). Our main motivation is to adapt a class of ES to FWI setting. ES's and all DFO algorithms are generally used only for relatively small problems (few hundreds of parameters in the best case). Thus for our numerical illustrations, we will try to represent the velocity model, as faithfully as possible, using the minimal number of parameters. We found out that the tested velocity model can be approximated using $n = 8 \times 8 \times 5 = 320$ parameters instead of $N = 225 \times 225 \times 70 = 3543750$. Using the 320 parameters, we are able to represent the velocity model and keep its main structure (i.e. the salt dome). The 320 parameters are computed using the real velocity model and the reduction procedure (see Figure 7.4). Figure 7.8 outlines an illustration of the obtained results using 320 parameters. As expected the magnification procedure using DCT transform (see Figures 7.8(g)- 7.8(i)), gives better results compared to the one based on Haar wavelets (see Figures 7.8(d)-7.8(f)). Although we use only few parameters to build the velocity model, the smoothing preserves the main specificity of the model, in particular the salt dome. Note that the 320 parameters, used to build the new velocity models (Figures 7.8(d) - 7.8(i)), are computed from the SEG/EAGE salt dome velocity model using the reduction procedure outlined in Figure 7.4.



(a) True model. (b) A vertical slice. (c) An horizontal slice.

(d) Duplicated model. (e) A vertical slice. (f) An horizontal slice.

(g) Magnified model. (h) A vertical slice. (i) An horizontal slice.

FIGURE 7.8: A 3D duplicated and magnified models of SEG/EAGE salt dome velocity model. The velocity models are built using $n = 8 \times 8 \times 5 = 320$, the original size of the true velocity model is of $N = 225 \times 225 \times 70 = 3543750$.

7.2.4 A parallel ES for acoustic full waveform inversion

Algorithm 7.2 presents an adaptation to the FWI setting of the globally convergent ES proposed in Algorithm 4.1. The monitoring of the quality of the sampling procedure is ensured by checking if the objective function has been sufficiently decreased. In fact, we proposed in Chapter 4 three different globally convergent ES versions: mean/mean, max/max and max/mean. The mean/mean version performed numerically the best among the other different versions. However, the incorporation of the mean/mean sufficient decrease condition requires an extra objective function evaluation $\mathcal{C}(m_{k+1}^{trial})$ at each iteration, where m_{k+1}^{trial} is the trial mean parent computed as the mean of the best μ generated velocity models. The mean/mean version is therefore corrupting the parallel nature of ES's. In fact, if one supposes that the offspring evaluation is performed at the same time using synchronized parallel clusters, the mean parent evaluation $\mathcal{C}(m_{k+1}^{trial})$ will force all these clusters to wait for the end of such evaluation to be able to restart a

new offspring generation. Therefore, the mean/mean version entails the parallel nature of our proposed ES.

Alternatively, the max/max version showed good performance (not as good as the mean/mean version) without the need of any extra objective function evaluation to impose the associated sufficient decrease condition. Consequently, the max/max version is more adapted to the parallel nature of ES than the mean/mean version. The updating of the weights (see Step 2 of Algorithm 4.1) to enforce the condition (4.1) was not activated for the two fold reasons: we wanted the least amount of changes in ES and since such an update of the weights did not seem to have a real impact on the results for the max/max version (see Section 4.2).

The proposed ES implementation will be a synchronized parallel optimizer composed of λ clusters (typically, the population size). Each cluster is composed of a group of processors, which is designed to evaluate the objective function (7.4). At a given iteration k, the clusters are synchronized and not activated until the new mean parent m_{k+1} is defined, depending on the iteration state (successful or not). The diagram in Figure 7.9 reports in detail our proposed parallel implementation of Algorithm 7.2. The implementation is as follows: A component [`Update Param.`] will be responsible for updating all the ES parameters (e.g. the distribution, the step length ...). In addition, it will launch asynchronously λ clusters represented in the diagram by the components [`Generate` m_i]. Each of these clusters generates a reduced velocity model based on the ES parameters and strategies. Once the velocity model is generated, the related cluster evokes the component [`Propagate` m_i].

The wave propagation simulation on each velocity model deals with all the p shots (many right-hand sides) at once. The [`Propagate` m_i] component is in fact an MPI (Message Passing Interface) process making use of processors and is responsible for discretizing and building the linear system to be solved (i.e. the forward problem) and to provide the information needed to evaluate the objective function $\mathcal{C}$ (7.4). The last component will just return the value of the objective function to the master. Once the master receives results from the λ clusters, it will choose the best results and return it to [`Update Param.`] to update the ES parameters and repeat the loop until a convergence criterion is achieved. At the end of each iteration, all the clusters send the simulated values to the [`Master`] component to decide either the iteration is successful or not and update the mean parent.

The propagation itself will behave as a black box process, hiding the complexity of the discretization and the solution of its respective linear system from ES. Also, the flexibility

Algorithm 7.2: An adaptation of the ES algorithm to FWI setting.

Initialization: Choose positive integers λ and μ such that $\lambda \geq \mu$. Select an initial $x_0 \in \mathbb{R}^n$, generate a velocity model $m_0 \in \mathbb{R}^N$ (using the magnification procedure) and evaluate $\mathcal{C}(m_0)$. Choose initial step lengths $\sigma_0, \sigma_0^{\text{ES}} > 0$ and initial weights $(\omega_0^1, \ldots, \omega_0^\mu) \in S$. Choose constants $\beta_1, \beta_2, d_{\min}, d_{\max}$ such that $0 < \beta_1 \leq \beta_2 < 1$ and $0 < d_{\min} < d_{\max}$. Select a forcing function $\rho(\cdot)$. Set $k = 0$.

Until some stopping criterion is satisfied:

1. **Generation of velocity models:** Generate λ velocity models $M_{k+1} = \{m_{k+1}^1, \ldots, m_{k+1}^\lambda\}$ using the magnification procedure based on the sample points $Y_{k+1} = \{y_{k+1}^1, \ldots, y_{k+1}^\lambda\}$ such that

$$y_{k+1}^i \;=\; x_k + \sigma_k d_k^i,$$

where $d_k^i \in \mathbb{R}^n$ is drawn from the distribution $\mathcal{C}_k$ and obeys $d_{\min} \leq \|d_k^i\|_2 \leq d_{\max}$, $i = 1, \ldots, \lambda$.

2. **Parent Selection:** Evaluate $\mathcal{C}(m_{k+1}^i)$, $i = 1, \ldots, \lambda$, and reorder the offspring points in $Y_{k+1} = \{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\lambda\}$ by increasing order: $\mathcal{C}(\tilde{m}_{k+1}^1) \leq \cdots \leq \mathcal{C}(\tilde{m}_{k+1}^\lambda)$. Select the new parents as the best μ offspring sample points $\{\tilde{y}_{k+1}^1, \ldots, \tilde{y}_{k+1}^\mu\}$, and compute their weighted mean

$$x_{k+1}^{trial} \;=\; \sum_{i=1}^{\mu} \omega_k^i \tilde{y}_{k+1}^i.$$

Magnify x_{k+1}^{trial} to obtain the velocity model m_{k+1}^{trial}.

3. **Imposing Sufficient Decrease:**

If $\mathcal{C}(\tilde{m}_{k+1}^\mu) \;\leq\; \mathcal{C}(m_k^\mu) - \rho(\sigma_k)$, then consider the iteration successful, set $x_{k+1} = x_{k+1}^{trial}$, $m_{k+1} = m_{k+1}^{trial}$, and $\sigma_{k+1} \geq \sigma_k$ (for example $\sigma_{k+1} = \max\{\sigma_k, \sigma_k^{\text{ES}}\}$). Set also $m_{k+1}^\mu = \tilde{m}_{k+1}^\mu$.

Otherwise, consider the iteration unsuccessful, set $x_{k+1} = x_k$, $m_{k+1} = m_k$ and $\sigma_{k+1} = \bar{\beta}_k \sigma_k$, with $\bar{\beta}_k \in (\beta_1, \beta_2)$. Set $m_{k+1}^\mu = m_k^\mu$.

4. **ES Updates:** Update the ES step length σ_{k+1}^{ES}, the distribution $\mathcal{C}_k$, and the weights $(\omega_{k+1}^1, \ldots, \omega_{k+1}^\mu) \in S$. Increment k and return to Step 1.

and the modularity of the propagator component is a key property, such that changing the chosen solver and/or the discretization nuances will not incur in any rewriting of ES implementation. MPI-2 has been used with the `MPI_COMM_SPAWN` interface that allows an MPI process to spawn a number of clusters. Each newly spawned cluster has a new `MPI_COMM_WORLD` intracommunicator that allows to launch easily the propagation simulations. The proposed ES implementation is portable and the propagator itself can be a standalone server. When the available cluster number is less than λ, one can launch many propagation simulations on the same cluster until we get the needed

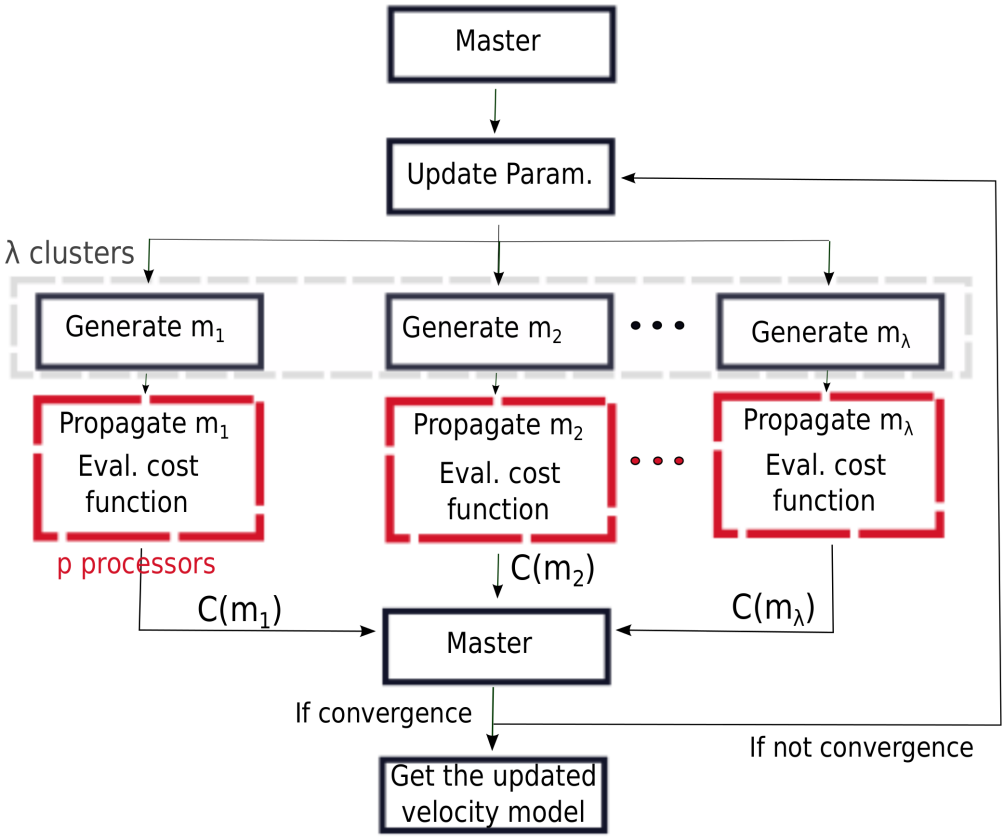


FIGURE 7.9: A parallel evolution strategy for full waveform inversion.

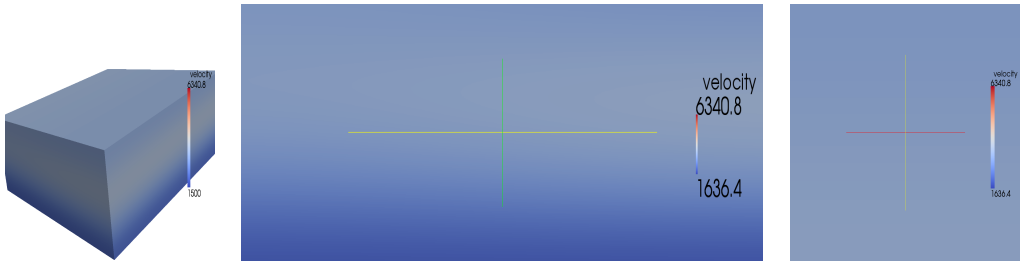
function evaluations.

7.3 Numerical experiments

7.3.1 Implementation details

We proposed an implementation of our parallel ES (see Figure 7.9) using FORTRAN03. FORTRAN03 is a FORTRAN compiler published in 2004, it is developed, for instance, in ifort (Intel FORTRAN Compiler), gfortran (GNU FORTRAN compiler) and other compilers. Using FORTRAN03, allawed us to implement an object oriented prototype code similar to the one proposed by [113] to solve the forward problem using the iterative solver proposed in [41]. Such choice was motivated by the fact that compared to FORTRAN90, the object oriented prototype in FORTRAN03 did not bring any slow down to the performance of the code and showed a speed up of 1% to 3.5% (see [113]).

For the propagation simulation, we use a simple scenario whereby the source excitation s(x) is supposed to be known (a Dirac function) and the observed data d^i_{obs} (i.e. seismograms) are generated from the propagating velocity model we are trying to invert (see Section 7.2.2). Only 16 sources are considered for our numerical simulation, the sources are uniformly distributed in a survey plan fixed at 500 meters of depth (10% of the exploration depth). Figure 7.10 reports the velocity model used as our initial point for the parallel evolution strategy in all our numerical simulations. The initial velocity model is built using the magnification procedure of two known velocity values, the first one is the velocity value on the bottom which is estimated as 3000 m/s and the second one is the value on the top estimated as 1500 m/s.



(a) The full model. (b) A vertical slice. (c) An horizontal slice.

FIGURE 7.10: The starting velocity model for the parallel evolution strategy.

In all the experiments we consider frequencies of 1 Hz, 2 Hz, and 3 Hz. With 320 velocity model parameters, we were able to represent the general aspects of the true velocity model (see Section 7.2.3). 320 unknown parameters for the ES is an acceptable number to explore, thus in all our experiments the search space of the implemented ES is of dimension $n = 8 \times 8 \times 5 = 320$. Our tests were performed on 2048 cores, the number of the cluster and the population size λ are adapted to the working frequency. For instance, for the 1 Hz case we used 256 clusters of 16 cores each. The population size λ was set to 512 meaning that each cluster ensures two objective function evaluations. Table 7.1 reports the distribution of the cluster number as well as the population size λ depending on the working frequency. One can notice that as far as the frequency range is increasing the number of cores dedicated to the objective function evaluation gets larger. In fact, the forward problem gets more complicated to solve as far as the working frequency f increases. When the number of the available clusters is less than the population size λ, we launch a fixed number of evaluations on the same cluster until we evaluate all the offspring population.

The other parameters are those of CMA-ES for unconstrained optimization (see [78]): $\mu = \text{floor}(\lambda/2)$, where floor$(\cdot)$ rounds to the nearest integer, and $\omega^i_0 = a_i/(a_1 + \cdots + a_\mu)$ and $a_i = \log(\lambda/2 + 1/2) - \log(i)$, $i = 1, \ldots, \mu$. The choices of the distribution $\mathcal{C}_k$ and of

Frequency	Problem size N	Number of clusters	Population size λ
1 Hz	$136 \times 136 \times 34$	256 (16 cores/cluster)	512 (2 evaluations/cluster)
2 Hz	$272 \times 272 \times 68$	64 (32 cores/cluster)	320 (5 evaluations/cluster)
3 Hz	$408 \times 408 \times 102$	32 (64 cores/cluster)	320 (10 evaluations/cluster)

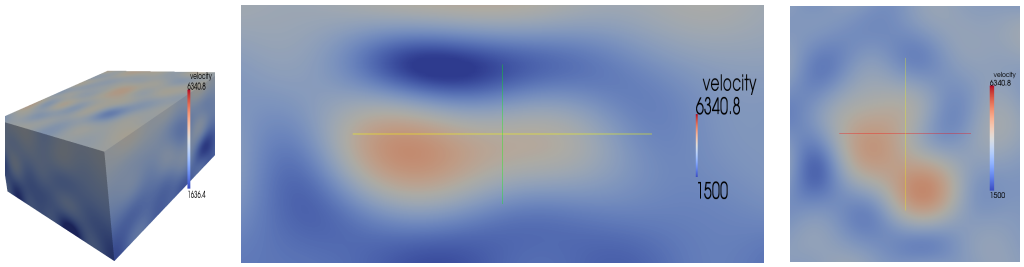
TABLE 7.1: The distribution of the clusters and the population size depending on the working frequency.

the update of σ_k^{ES} also followed CMA-ES for unconstrained optimization (see [78]). The forcing function selected was $\rho(\sigma) = 10^{-4}\sigma^2$. To reduce the step length in unsuccessful iterations we used $\sigma_{k+1} = 0.5\sigma_k$ which corresponds to setting $\beta_1 = \beta_2 = 0.5$. The initial step size σ_0 is estimated to half of the difference between the velocity value on the bottom which is estimated as $3000m/s$ and the second one is the value on the top estimated as $1500m/s$.

7.3.2 Numerical Results

All our numerical experiments are tested on a 2048 CPU Sandy Bridge machine. We had no maximal computational budget concerning the function evaluations as far as the computation elapsed time does not exceed 1 day. Thus, a run can not exceed 24 hours otherwise the inversion procedure will be stopped.

Figure 7.11 reports a graphical representation of the inverted velocity model considering a frequency of 1 Hz. The obtained results in this case can be seen as good, since it represents a smooth version of the true velocity model we are looking for. For the 1 Hz case, we are able to invert the general structure of the regarded velocity model, in particular the salt dome structure.



(a) The inverted model. (b) A vertical slice of the inverted model. (c) An horizontal slice.

FIGURE 7.11: Inversion results for the Salt dome velocity model using $n = 320$ parameters. The working frequency is of 1 Hz.

After 278 iterations, the inversion procedure is stopped due to the maximal time on the machine (24 hours). Figure 7.12 outlines the objective function evaluation at the best

population point of each generation. The variation of the objective function is more significant only in the early stages. Such behavior is due to the sufficient decrease condition which monitors the quality of the sampling procedure and ensures the convergence to a stationary point (see Section 4.1.2).

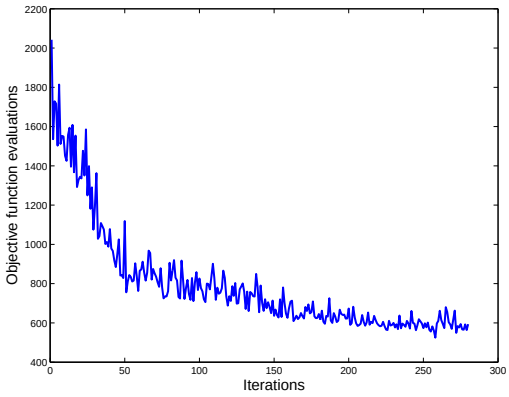
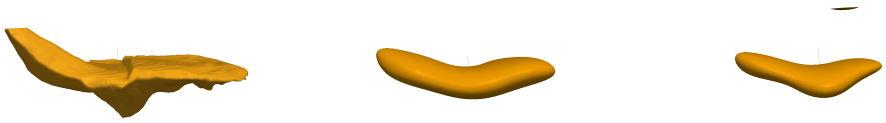


FIGURE 7.12: Objective function evaluation at the best population point for the first 278 iterations of the parallel evolution strategy.

The parallel ES leads to a new velocity model that approximates the general structure of the true velocity model. Figure 7.13 reports a graphical representation of the interior of three velocity models: (a) the true velocity model (Figure 7.13(a)), (b) its approximation using 320 parameters (Figure 7.13(b)), (c) and the inversion results using $n = 320$ unknowns (Figure 7.13(c)). The approximation (built using 320 parameters selected from the true velocity model) can be seen as the velocity model we target to obtain using our inversion procedure and with $n = 320$ unknowns. The inverted velocity model is similar to the approximation we are looking for, in particular the salt dome.



(a) The slat dome of the true velocity model.

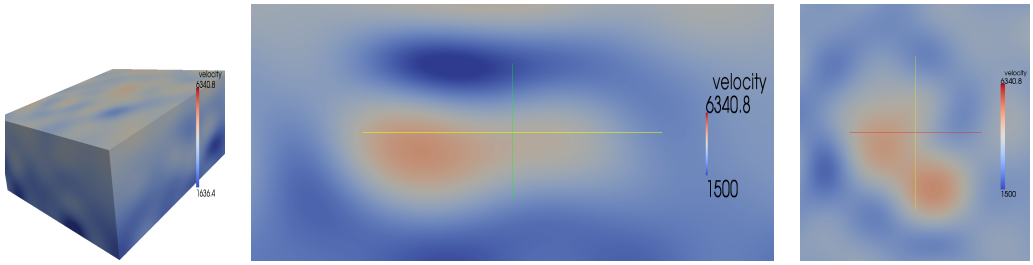
(b) The targeted salt dome using $n = 320$ parameters.

(c) The obtained salt dome using $n = 320$ parameters.

FIGURE 7.13: Graphical representation of the salt dome of three velocity models: the true velocity salt dome (Figure 7.13(a)), the approximated one using 320 parameters (Figure 7.13(b)), and the inverted velocity model (Figure 7.13(c)). Only the points of the models which have velocity equal or larger than 3500 m/s are shown (to delineate the structure of the dome of salt).

Figure 7.14 reports inverted velocity models for different frequency range using $n = 8 \times 8 \times 5 = 320$ parameters. As far as the working frequency f increases (from 1 Hz to 3 Hz),

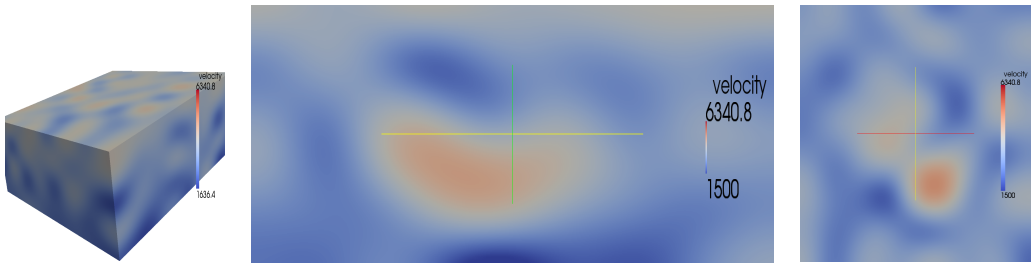
the inversion result is getting less accurate and far from being a good approximation of the targeted velocity model (see Figure 7.13(b)). The explanation of such results is two fold: (a) the objective function becomes more and more noisy and multi-modal as far as the frequency increases [155], (b) the computational cost of the objective function increases and demands more computational resources to be evaluated. In fact unlike 1 Hz frequency case where 278 iterations were performed, only 50 iterations (resp. 22 iterations) are performed in the 2 Hz (resp. 3 Hz) frequency experiments. The small number of iterations explains the inversion results obtained of such range of frequencies. For the moment, we are working on adapting the implemented code to include a restart option, such option will unable the computational cost in CPU to exceed 24 hours. Consequently, we will be able to get asymptotic inversion results for high frequencies.



(a) The inverted model using a frequency of 1 Hz.

(b) A vertical slice.

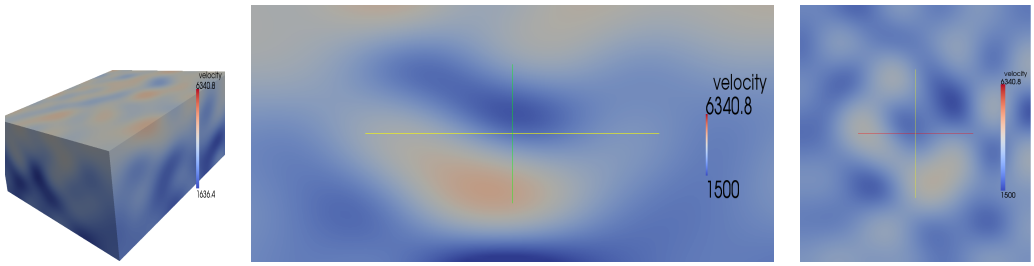
(c) An horizontal slice.



(d) The inverted model using a frequency of 2 Hz.

(e) A vertical slice.

(f) An horizontal slice.



(g) The inverted model using a frequency of 3 Hz.

(h) A vertical slice.

(i) An horizontal slice.

FIGURE 7.14: Comparison of the inversion results for the Salt dome velocity model using $n = 320$ parameters for different range of frequencies (1 Hz, 2 Hz and 3 Hz).

7.4 Conclusions

The main contribution of this chapter was to show a possible way to adapt ES's to the FWI setting. For that purpose, we proposed a new parametrization of the regarded problem, by being able to represent the velocity model as faithfully as possible, while limiting the number of parameters needed, since each additional parameter is an additional dimension to explore. A highly parallel ES adapted to FWI setting was proposed and validated.

We showed on an academic velocity model the efficiency of the new proposed parametrization. In fact, we were able to reconstruct a good approximation of this velocity model using only few parameters. The proposed parallel implementation was tested and validated using the new parametrization of the regarded problem, the initial obtained results showed that great improvement can be expected in the automation of the FWI procedure.

Chapter 8

Conclusions & Perspectives

8.1 Conclusions

This thesis has contributed to the research area of Evolution Strategies (ES's) by addressing the following challenges:

(i) Modifying a class of ES's, for unconstrained optimization, to rigorously achieve a form of global convergence under reasonable assumptions.

(ii) Proposing a new approach to extend a class of ES's to handle general constrained optimization problems. The proposed algorithm is designed to be globally convergent regardless of the starting points.

(iii) Showing a new possible way to incorporate surrogate quadratic models in the proposed ES to achieve a better performance.

(iv) Proposing an adaptation of our proposed ES to the acoustic full-waveform inversion related to Earth imaging problem.

The challenge (i) was addressed in Chapter 4 by showing how to modify a large class of ES's so that they converge to stationary points without any assumption on the starting point [58]. We proposed different ways of imposing sufficient decrease for which global convergence holds under reasonable assumptions. The so-called mean/mean version, where the step size is reduced whenever the objective function value of the weighted mean of the best trial offspring does not sufficiently reduce the objective value at the current weighted mean, has emerged as the best modified version in our numerical experiments. Moreover, we have shown that such an improvement in efficiency came without

weakening significantly the performance of the underlying method in the presence of several local minimizers.

The challenge (ii) was addressed in Chapter 5 by extending our proposed ES to handle general constrained optimization [59]. For non-relaxable constraints, we proposed two feasible approaches. In a first approach, feasibility is first enforced by a barrier function and the objective function is then evaluated directly at the feasible generated points. A second approach projects first all the generated points onto the feasible domain before evaluating the objective function. The relaxable constraints were handled using a merit function approach. Compare to existing algorithms, the obtained numerical results were interesting for both relaxable and unrelaxable constraints and they confirm the competitiveness of our solver.

The challenge (iii) was addressed in Chapter 6 where at the beginning of each iteration of our proposed ES, a search step was token. For that purpose, a surrogate quadratic model of the objective function f was minimized in a certain region using previously evaluated points. Our hybrid algorithm was designed to satisfy the convergence analysis of our globally convergent ES. The numerical experiments have shown that incorporating local models improved the performance of our ES in both unconstrained and constrained optimization problems.

Finally, the challenge (iv) was addressed in Chapter 6 by using the proposed ES to find a starting velocity model for the acoustic full-waveform inversion [57]. We adapted our ES to the problem settings. A subspace approach was used for the parametrization of the considered problem. A highly parallel implementation of our modified ES was proposed. The obtained results provide a great improvement to known solutions of this problem.

8.2 Perspectives

Several extensions for the present research can be mentioned. In the proposed theoretical analysis, we assumed the absence of noise on the objective function. However, in practice ES's are more designed to solve simulation optimization problems where one has a wide range of uncertainty. An example of such problems occurs, for instance, when the objective function involves inaccurate solutions of a PDE as in the acoustic full-waveform inversion case (e.g. with a truncated iterative solvers, with a discretization size, ...). Inspired by recent works [31, 106, 115], a generalization of our theoretical analysis to include uncertainty on the objective function is of interest for future work.

The theoretical analysis in Chapters 4 and 5 would be done deterministically, as if we were considering a single realization of a stochastic algorithm. Most likely, in our case

one could also analyze ES's when the sampling points are regarded as random variables, and to investigate such a framework towards almost-sure global convergence properties. A possible way to tackle this approach is by using techniques similar to trust-region methods based on probabilistic models developed recently [27] (also used in direct search for probabilistic descent [73]).

For the modified ES proposed in Chapter 4, the version mean/mean, consisting of applying sufficient decrease directly to the weighted mean of the new parents, has been shown to yield global convergence without any convexity like assumption and to numerically perform the best among the tested versions. However, the incorporation of such sufficient decrease condition entails the parallel nature of ES's since an extra objective function evaluation at the trial mean parent is needed at each iteration. A possible way to overcome such inconvenient is by not using the sufficient decrease condition in all iterations, but only at a certain probability. Such modification as well as its theoretical impact are to be investigated in future work.

The performance of the proposed algorithm, for relaxable constrained optimization problems (see Section 5.3.2), was validated only by looking at individual results obtained for each test problem. Adapting data and performance profile test strategies to relaxable constraints would be more suitable to quantify the performance of our proposed algorithm compared to other existing ones. The use of these profiles require, in turn, an adaptation of the convergence test to take into account a fixed tolerance on the constraints violation (see Section 4.2.3). Future investigations on the optimal way to adapt data and performance profiles, in particular efficient convergence tests, are needed to quantify the performance of the algorithm for relaxable constrained optimization problems.

For general linear constraints, the incorporation of the quadratic models in the search step of the proposed algorithm did not lead to any significant performance improvement. Therefore, it would be also interesting to further explore an efficient search step procedure for such constrained setting.

The purpose of the incorporation of ES's in the inversion procedure of full-waveform inversion was to find a good starting point without the need for sophisticated a priori knowledge on the background velocity model. The validation of such statement is not entirely addressed in Chapter 7, in the sense that we did not test the full-waveform inversion using the velocity model obtained by the proposed ES. The next step will be to validate the obtained results using a gradient-based method.

Moreover, for the acoustic full-waveform inversion problem, the proposed parallel implementation was tested using a simple scenario whereby the source excitations were

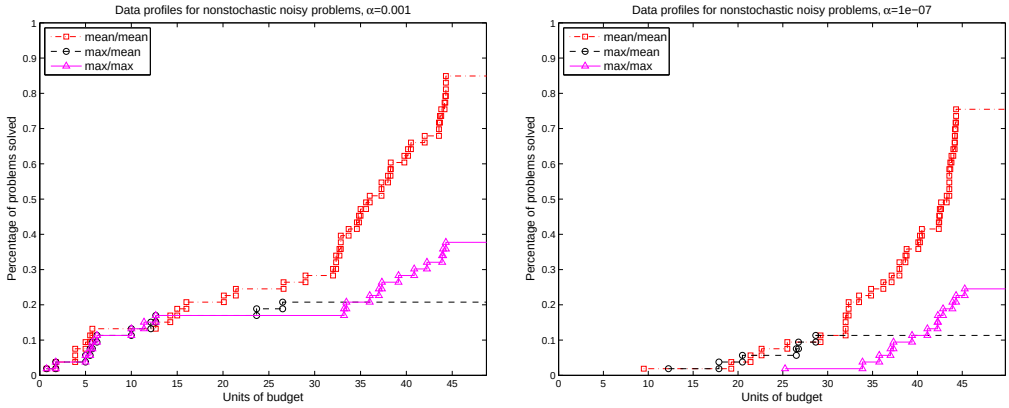
supposed to be known and the observed data (i.e. seismograms) was generated using the propagating velocity model which we are trying to invert. Such assumptions are not realistic for the following reasons: (a) the source excitation is generally unknown and it must be included as an unknown of the problem, (b) the observed data is generally given by geophones situated on the surface of the exploration domain. A more realistic test is to be investigated in future work.

In this thesis, we tackled large scale inverse problems by stochastic optimization via model reduction techniques. Under appropriate assumptions, the model reduction procedure, proposed in this thesis, can be generalized to cover other geoscience applications. The developed methods can be applied to other geoscience optimization problems (e.g. well placement, formation-evaluation inversion, ...), since we believe that many geoscience problems could be successfully handled with the algorithms proposed in this work.

Appendix A

Data & Performance Profiles Results

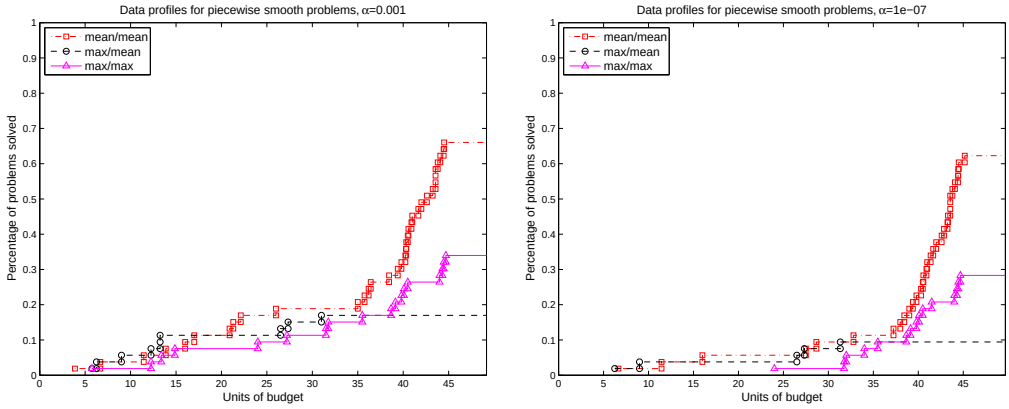
Figures A.1 to A.6 outline a comparison between the three modified versions of CMA-ES (mean/mean, max/max, and max/mean) using data and performance profiles. In Section 4.2.4, we reported only comparison results for the class of smooth problems. The following appendix report the remaining comparison results for the other class of problems (meaning nonstochastic noisy, piecewise smooth, and stochastic noisy problems). The obtained results followed a very similar trend in the sense that the mean/mean version emerges as the best one for all the problem tested.



(a) Accuracy level of 10^{-3}.

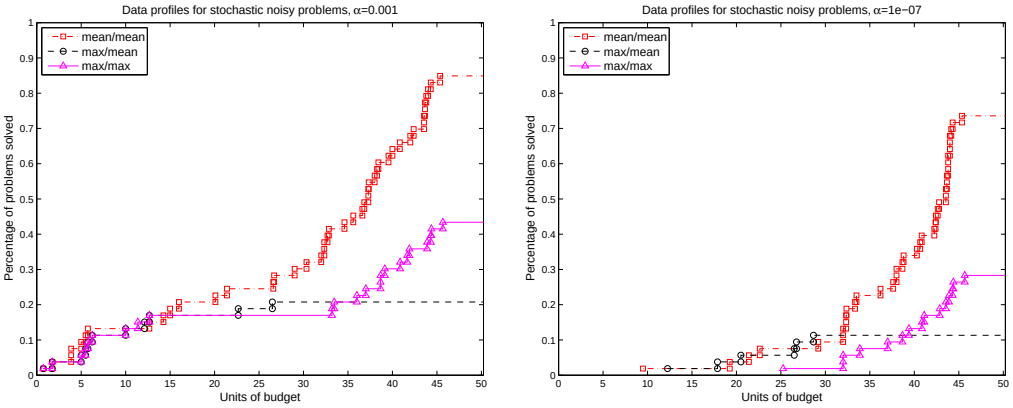
(b) Accuracy level of 10^{-7}.

FIGURE A.1: Data profiles computed for the set of nonstochastic noisy problems, considering the two levels of accuracy, 10^{-3} and 10^{-7} (for the three modified versions).



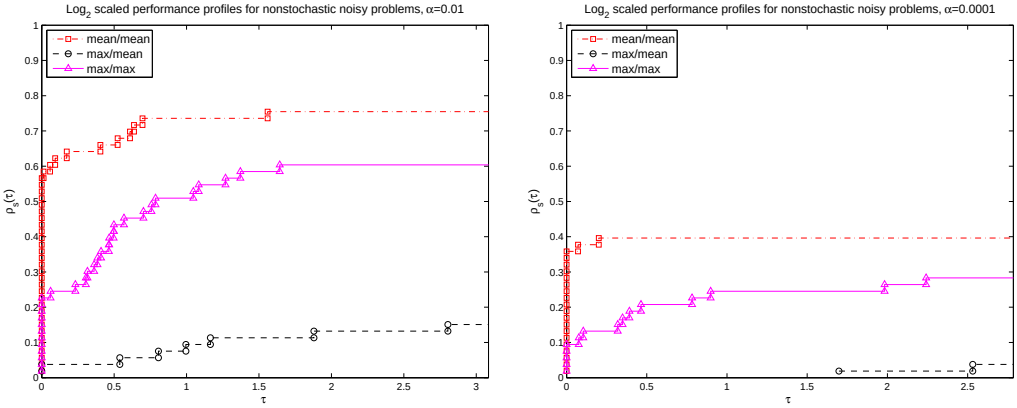
(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE A.2: Data profiles computed for the set of piecewise smooth problems, considering the two levels of accuracy, 10^{-3} and 10^{-7} (for the three modified versions).



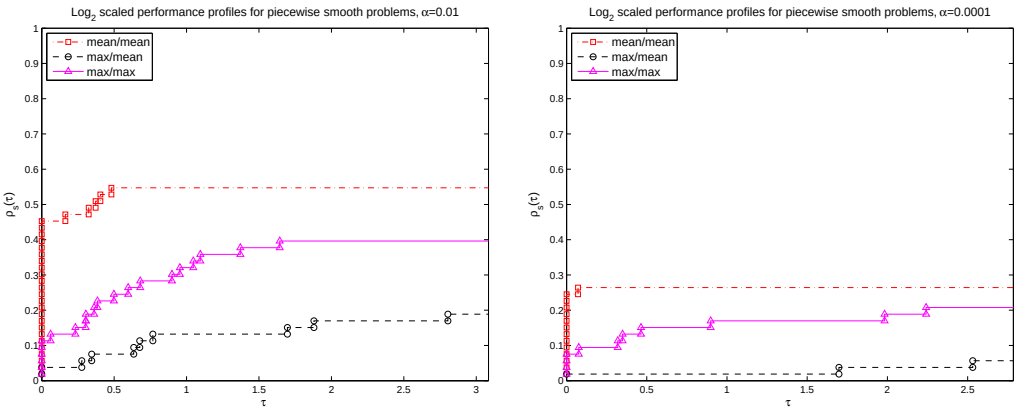
(a) Accuracy level of 10^{-3}. (b) Accuracy level of 10^{-7}.

FIGURE A.3: Data profiles computed for the set of stochastic noisy problems, considering the two levels of accuracy, 10^{-3} and 10^{-7} (for the three modified versions).



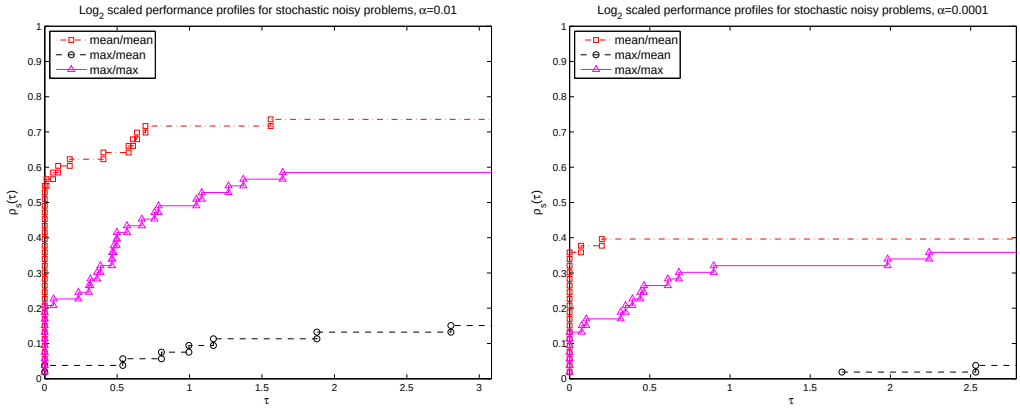
(a) Accuracy level of 10^{-2}. (b) Accuracy level of 10^{-4}.

FIGURE A.4: Performance profiles computed for the set of nonstochastic noisy problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4} (for the three modified versions).



(a) Accuracy level of 10^{-2}. (b) Accuracy level of 10^{-4}.

FIGURE A.5: Performance profiles computed for the set of piecewise smooth problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4} (for the three modified versions).



(a) Accuracy level of 10^{-2}.

(b) Accuracy level of 10^{-4}.

FIGURE A.6: Performance profiles computed for the set of stochastic noisy problems with a logarithmic scale, considering the two levels of accuracy, 10^{-2} and 10^{-4} (for the three modified versions).

Appendix B

Test Results

The results of bound-constrained and linear-constrained testing are depicted in the tables below. The first three columns of all the tables describe the test problems used from [164, 165] (the problem name, the dimension n and the objective function value at the global minimum f^*), the other columns explicit the optimal objective function values found by each solver. We ran the numerical experiments using a maximal budget of 1500 function evaluation.

Name	n	f^*	ES-LC-B	ES-LC-P	PSWARM	CMA-ES	MCS	BC-DFO
ack	10	$-4.44089e-16$	$-4.44089e-16$	$-4.44089e-16$	0.628445	0.582773	0.499069	0.125355
ap	2	-0.35239	-0.352386	-0.352386	-0.352386	-0.332411	-0.352386	-0.352386
bf1	2	$-5.55112e-17$	$-5.55112e-17$	$-5.55112e-17$	$-5.55112e-17$	$-5.55112e-17$	$-5.55112e-17$	$-5.55112e-17$
bf2	2	0	0	0	0	0	0	0
bhs	2	-3.4285	-3.42849	-3.42849	-3.42849	-2.60574	-3.42849	-2.21048
bl	2	0	$2.23592e-11$	$5.10822e-14$	$4.24679e-11$	$6.20803e-14$	0	$3.79442e-28$
bp	2	0.397887	0.397887	0.397887	0.397887	0.397887	0.397887	0.397887
cb3	2	0	0	0	0	0	0	0
cb6	2	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163	-1.03163
cm2	2	-0.2	-0.2	-0.2	-0.2	-0.2	-0.2	-0.2
cm4	4	-0.4	-0.4	-0.4	-0.4	-0.4	-0.4	-0.4
da	2	-24777	-24776.5	-24776.5	-24776.5	-14865.9	-24776.5	0
em_10	10	-8.40797	-6.23631	-5.85721	-6.42058	-5.29973	-6.68539	-8.27162
em_5	5	-4.6877	-4.4935	-3.69458	-3.5116	-3.48061	-3.61152	-4.49589
ep	2	-1	-1	-1	-1	-0.700024	-1	-1
exp	10	-1	-1	-1	-1	-1	-1	-1
fls	2	-2.02181	-2.02181	-2.02181	-2.02181	-2.02181	-2.02181	-0.994945
fr	2	0	$3.70128e-09$	$5.8461e-15$	19.7028	65.6759	65.6762	65.6759
fx_10	10	-10.2088	-1.14662	-1.47569	-1.13584	-1.09341	-1.47976	-3.27511
fx_5	5	-10.4056	-3.47742	-1.83376	-1.97401	-2.36457	-2.70097	-1.57965
gp	2	3	3	3	3	13.8	3	3
grp	3	0	$1.4757e-15$	$1.72892e-11$	$1.42888e-11$	$4.3576e-15$	$1.72346e-08$	$3.15544e-26$
gw	10	0	0	0	0	0	0	0
h3	3	-3.8628	-3.86278	-3.86278	-3.86278	-3.86278	-3.86278	-3.08976

TABLE B.1: Results from comparison of the solvers on bound-constraind problems (average of 10 runs for stochastic solvers)- Part 1 .

Name	n	f^*	ES-LC-B	ES-LC-P	PSWARM	CMA-ES	MCS	BC-DFO
h6	6	-3.3224	-3.32237	-3.20312	-3.2984	-3.30961	-3.32237	-3.20316
hm	2	0	$4.67413e-08$	$4.65101e-08$	$4.68394e-08$	$4.65101e-08$	$4.65101e-08$	1.03163
hm1	1	0	$2.17195e-10$	$2.1719e-10$	$1.7862e-11$	$1.26249e-13$	$1.97215e-31$	0
hm2	1	-4.81447	-4.81447	-4.81447	-4.81447	-0.96288	-4.81446	0
hm3	1	-2.27	-2.26754	-2.26754	-2.26754	-2.1406	-2.26754	0
hm4	2	0	$1.23885e-10$	$1.45797e-13$	1.55818	$2.56643e-15$	$6.7576e-20$	15.5818
hm5	3	0	0	0	0	0	0	0
hsk	2	-2.34581	-2.34581	-2.34581	-2.34581	-1.74583	-2.34581	-1.12779
hv	3	$1.4201e-40$	$2.00938e-10$	$2.19337e-09$	$4.91066e-05$	$3.50567e-14$	$1.4201e-40$	25.9911
ir1	3	0	0	0	0	0	0	0
ir2	2	0	$3.06391e-10$	$1.03849e-13$	0.00157914	$9.19098e-15$	$2.64977e-20$	$8.24757e-22$
ir4	30	0	0.0519641	0.0519641	0.0519641	0.0519641	0.0519641	0.0519641
ir5	2	0.001996	0.00199969	0.0019995	0.0019985	0.00199969	0.00199969	0.00199969
kl	4	0.000307486	0.00030758	0.000307486	0.000330504	0.000307487	0.000307486	0.000307486
ks	1	0	0	0	0	0	0	0
lj1_38	114	3114.51	$2.33302e+06$	241023	15289.4	38347.4	$1e+09$	$1e+09$
lj1_75	225	523060	$3.19296e+08$	$6.2303e+06$	$1.33632e+06$	$4.58214e+06$	$1e+09$	$1e+09$
lj1_98	294	$1.83581e+06$	$2.53341e+09$	$2.55844e+07$	$1.10237e+07$	$5.41124e+07$	$1e+09$	$1e+09$
lj2_38	114	4227.85	$2.33312e+06$	241271	15879.6	43409.7	$1e+09$	$1e+09$
lj2_75	225	604764	$3.19297e+08$	$6.23102e+06$	$1.09771e+06$	$5.01557e+06$	$1e+09$	$1e+09$
lj2_98	294	$2.25008e+06$	$2.53341e+09$	$2.55858e+07$	$8.61482e+06$	$6.27853e+07$	$1e+09$	$1e+09$
lj3_38	114	51079.6	$1.22305e+09$	$2.19352e+06$	766845	$5.03627e+06$	$1e+09$	$1e+09$
lj3_75	225	$8.00081e+07$	$3.65358e+12$	$1.01587e+10$	$6.03574e+08$	$9.9765e+08$	$1e+09$	$1e+09$
lj3_98	294	$1e+09$	$6.98651e+13$	$5.10457e+10$	$8.07316e+09$	$1e+09$	$1e+09$	$1e+09$

TABLE B.2: Results from comparison of the solvers on bound-constraind problems (average of 10 runs for stochastic solvers)- Part 2 .

Name	n	f^*	ES-LC-B	ES-LC-P	PSWARM	CMA-ES	MCS	BC-DFO
lm1	3	0	$1.10763e-09$	$5.57824e-11$	$6.28474e-07$	$2.58288e-14$	$1.57851e-31$	$2.37534e-13$
lm2_10	10	0	0.0876906	0.0161266	0.00354416	0.0417907	$1.34969e-31$	0.0110986
lm2_5	5	0	0.0444938	0.06549	$1.34969e-31$	0.00853277	$1.34969e-31$	0.297625
lv8	3	0	$3.76424e-11$	$7.13608e-13$	$4.89571e-10$	$2.20881e-14$	$1.49966e-32$	$2.3585e-26$
mc	2	−1.91322	−1.91322	−1.91322	−1.91322	−1.91322	−1.91322	−1.91322
mcp	4	0	$8.3558e-10$	$2.48537e-09$	$2.76852e-14$	$8.46032e-14$	$7.52392e-17$	$6.92197e-13$
mgp	2	−1.297	−1.29695	−1.29695	−1.29695	−1.28138	−1.29695	−1.29695
mgw_10	10	0	0	0	0	0	0	0
mgw_2	2	0	0	0	0	0	0	0
mgw_20	20	$-3.55271e-15$	$-3.55271e-15$	$-3.55271e-15$	$-3.55271e-15$	$-3.55271e-15$	$-3.55271e-15$	$-3.55271e-15$
ml_10	10	−0.965	$-5.03947e-321$	$-1.2838e-111$	−0.0896697	$-2.1722e-27$	$-2.35441e-23$	$-9.29196e-298$
ml_5	5	−0.965	$-4.61423e-50$	$-3.3648e-56$	−0.3179	$-6.71071e-112$	−0.806	$-3.57988e-185$
mr	3	$4e-05$	0.0019131	0.00190027	0.00634313	0.00609367	0.0596046	0.120085
mrp	2	0	0.00741539	0.00741539	$2.30571e-09$	0.00519077	$5.85521e-22$	$6.18487e-19$
ms1	20	5.58379	8.54762	8.22735	9.04108	9.42908	118.737	$1e+09$
ms2	20	13.935	14.0129	15.1745	15.7835	19.0253	285.894	$1e+09$
nf2	4	0	0.0121389	0.184693	0.0872574	0.0130726	$1.43243e-06$	$7.45769e-05$
nf3_10	10	−210	−209.478	−209.731	−203.417	−209.326	−210	−210
nf3_15	15	−665	−445.603	−378.426	−489.492	−482.99	−665	−665
nf3_20	20	−1520	−295.177	−222.102	−875	−382.666	−897.54	−1520
nf3_25	25	−2900	−243.857	−237.519	−1358.91	−163.263	−1203.05	−2900
nf3_30	30	−4930	−47.9737	−39.6664	−1921.88	−100.547	−1166.35	−4930
osp_10	10	−1.1438	$-1.8452e-07$	$-1.7843e-07$	$-1.10174e-05$	−0.030832	$-4.89477e-31$	$-1.46314e-11$
osp_20	20	−1.1438	$-9.69548e-12$	$-6.49482e-13$	$-1.53037e-12$	$-7.33036e-05$	$-3.06537e-50$	$-2.53022e-16$

TABLE B.3: Results from comparison of the solvers on bound-constraind problems (average of 10 runs for stochastic solvers)- Part 3 .

Name	n	f^*	ES-LC-B	ES-LC-P	PSWARM	CMA-ES	MCS	BC-DFO
plj_38	114	4286.56	$2.33307e+06$	242398	14740.9	48377.6	$1e+09$	$1e+09$
plj_75	225	471921	$3.19296e+08$	$6.23112e+06$	$1.33326e+06$	$6.24924e+06$	$1e+09$	$1e+09$
plj_98	294	$3.35843e+06$	$2.53341e+09$	$2.55862e+07$	$5.31535e+06$	$6.08277e+07$	$1e+09$	$1e+09$
prd	2	0.9	0.9	0.9	0.9	0.9	0.9	0.9
ptm	9	0	191.592	221.599	129.871	196.47	106.427	167.199
pwq	4	0	0	0	0	0	0	0
rb	10	0	8.60723	8.16482	7.77405	7.92561	2.34966	0.699086
rg_10	10	0	0	0	0	0	0	0
rg_2	2	0	0	0	0	0	0	0
s10	4	−10.5364	−3.83543	−2.42734	−7.5744	−5.12848	−10.5364	−5.12848
s5	4	−10.1532	−10.1532	−5.0552	−5.09827	−5.0552	−10.1532	−5.0552
s7	4	−10.4029	−3.7243	−5.08767	−8.11007	−5.08767	−10.4029	−5.08767
sal_10	10	0	0	0	1.23612	0.201177	$7.63753e-05$	0.0998733
sal_5	5	0	0	0	0.399955	0.131529	$1.35237e-06$	0.0998733
sbt	2	−186.731	−186.731	−186.731	−186.731	−140.879	−186.731	−186.731
sf1	2	0	0	0	0.00874432	0.00766049	0.0372241	0
sf2	2	0	$2.45961e-09$	$2.45961e-09$	0.124129	0.102956	0.312435	1.0127
shv1	1	−1	−1	−1	−1	−0.553195	−0.999996	0.841471
shv2	2	0	0	0	0	0	0	0
sin_10	10	−3.5	−0.997113	−0.999626	−3.27758	−0.999481	−3.5	−0.500032
sin_20	20	−3.5	$-1.50398e-05$	−0.949737	−1.80165	−0.416575	−1.00196	−0.11457
stg	1	0	0	0	0	0	0	0
st_17	17	0	$1.33425e+07$	$2.72246e+06$	156317	$1.1722e+06$	612661	35070.1
st_9	9	0	17.6671	41.6756	155.064	144.136	2.8266	39.1974

TABLE B.4: Results from comparison of the solvers on bound-constraind problems (average of 10 runs for stochastic solvers)- Part 4 .

Name	n	f^*	ES-LC-B	ES-LC-P	PSWARM	CMA-ES	MCS	BC-DFO
swf	10	-4189.83	-1899.25	-1543.19	-3655.89	-39.4503	-4071.39	$1e+09$
sz	1	-12.0312	-12.0312	-12.0312	-12.0312	-9.13785	-12.0312	-3.60801
szzs	1	-1.60131	-1.60131	-1.60131	-1.60131	-0.798275	-1.60131	-1.60131
wf	4	0	0.24724	5.12165	0.756791	0.923078	$3.83443e-19$	$4.15548e-26$
xor	9	0.867827	0.882538	0.880822	0.872623	0.875512	0.880278	0.879814
zkv_10	10	0	0	0	0	0	26.0722	0
zkv_2	2	0	0	0	0	0	$1.21861e-43$	0
zkv_20	20	0	0	0	0	0	320.055	0
zkv_5	5	0	0	0	0	0	$3.7638e-28$	0
zlk1	1	-1.91	-1.21598	-1.21598	-1.90596	-1.21136	-1.90596	-1.21598
zlk2a	1	-1.125	-1.125	-1.11725	-1.125	-0.747352	-1.125	-1.125
zlk2b	1	-1.125	-1.125	-1.125	-1.11258	-1.02635	-1.125	-1.125
zlk3a	1	-1	-0.999999	-1	-1	0	-1	0
zlk3b	1	-1	-1	-1	-1	0	-1	0
zlk3c	1	-1	-1	-1	-1	-0.3	-1	0
zlk4	2	0.397888	0.397888	0.397888	0.397888	0.397888	0.397888	0.397888
zlk5	3	-3.86278	-3.86278	-3.86278	-3.86278	-3.78548	-3.86278	-3.08976
zzs	1	-0.824239	-0.824239	-0.824239	-0.824239	-0.659392	-0.824235	-0.824239

TABLE B.5: Results from comparison of the solvers on bound-constraind problems (average of 10 runs for stochastic solvers) - Part 5 .

Name	n	m	f^*	ES-LC-B	ES-LC-P	PSWARM
oet1	3	402	0.538232	0.538239	0.538433	0.550358
oet3	4	402	0.00450873	0.00814026	0.996249	0.784257
avgasa	6	18	−4.1687	−4.16857	−4.1687	−4.1687
avgasb	6	18	−4.13282	−4.13274	−4.13267	−4.13282
bc4	127	254	12.9296	386.172	14.5377	396.077
biggsc4	4	21	−24.5	−24.4999	−24.4995	−24.5
bunnag1	3	7	0.111111	0.111113	0.111111	0.111112
bunnag2	4	10	−6.40521	−6.40496	−6.40521	−6.40521
bunnag3	5	11	−16.3693	−16.3632	−16.3693	−16.3681
bunnag4	6	13	−213.047	−212.433	−213.047	−213.047
bunnag5	6	16	−11.005	−10.3921	−10.9994	−11
bunnag6	10	31	−268.014	−254.582	−266.245	−247.188
bunnag7	10	25	−39	−28.8473	−32.4634	−27.6222
bunnag8	20	50	−394.751	−104.073	−79.1346	−220.173
bunnag9	20	50	−884.751	−600.765	−524.419	−664.959
bunnag10	20	50	−8695.01	−2418.84	−2000.21	−4265.74
bunnag11	20	50	−754.751	−558.936	−584.181	−611.203
bunnag12	20	50	−4105.28	−1212.8	−1412.35	−2686.53
bunnag13	20	50	49318	192764	547663	150752
ex2_1_1	5	11	−17	−17	−17	−17
ex2_1_10	20	30	57637.5	196582	210572	112476
ex2_1_2	6	13	−213	−213	−213	−213
ex2_1_3	13	29	−15	−15	−15	−15
ex2_1_4	6	14	−11	−11	−11	−11

TABLE B.6: Results from comparison of the solvers on linear-constraind problems (average of 10 runs for stochastic solvers)- Part 1 .

Name	n	m	f^*	ES-LC-B	ES-LC-P	PSWARM
ex2_1_5	10	31	-268.014	-252.519	-265.283	-254.306
ex2_1_6	10	25	-39	-39	-39	-39
ex2_1_7	20	30	-4049	-1207.99	-3447.36	-3668.73
expfita	5	21	0.00150644	0.497078	0.00150644	0.0176753
expfitb	5	101	0.00560556	0.00560556	0.545945	1.08841
expfitc	5	501	0.0414654	5.0458	0.0414654	3.89498
fir_linear	11	78	14.1549	80.6529	78.2801	36.393
g01	13	32	-15	0	-14.9655	-15
genocop07	6	14	-213	-212.545	-213	-210.727
genocop09	3	11	-2.47143	-2.4714	-2.47143	-2.47142
genocop10	4	10	-4.52837	-4.52833	-4.52835	-4.52837
genocop11	6	17	-11	-10.3567	-10.9952	-11
goffin	51	50	0	0	0	71.287
gtm	59	115	636.321	769.851	751.711	662.786
hatfldh	4	21	-24.5	-24.4998	-24.4997	-24.5
himmelbi	100	112	$7.25949e+23$	$8.78169e+50$	$2.65669e+50$	$2.05289e+42$
hs021	2	5	-99.96	-99.96	-99.96	-99.96
hs024	2	4	-1	-0.999991	-1	-0.999996
hs035	3	4	0.111111	0.111114	0.111112	0.111112
hs036	3	7	-3456	-3300	-3300	-3300
hs037	3	7	-3456	-3456	-3456	-3456
hs044	4	10	-15	-12.9986	-15	-13.8
hs076	4	7	-4.68182	-4.6817	-4.68182	-4.68181
hs086	5	11	-32.3487	-32.3275	-32.3482	-32.3486

TABLE B.7: Results from comparison of the solvers on linear-constraind problems (average of 10 runs for stochastic solvers)- Part 2 .

Name	n	m	f^*	ES-LC-B	ES-LC-P	PSWARM
hs118	15	59	664.82	686.615	688.243	668.125
hs21mod	7	9	−95.96	−95.96	−95.9597	−95.96
hs268	5	5	0.173975	0.444779	0.173975	3.03662
hs35mod	2	3	0.25	0.25	0.25	0.25
hs44new	4	9	−15	−12.9996	−15	−14
hubfit	2	2	0.0168935	0.0168935	0.0168935	0.0168939
Ji1	3	7	−4.0907	−4.0907	−4.0907	−4.0907
Ji2	3	5	−3.00292	−3.00246	−3.00292	−3.00292
Ji3	2	3	−5.99989	−5.99201	−5.99966	−5.99663
ksip	20	201	0.829106	0.937146	1.04348	1.34824
liswet1	202	200	50.9896	50.9896	50.9896	50.9896
liswet10	202	200	51.3597	51.3597	51.3597	51.3597
liswet11	202	200	51.3321	51.3321	51.3321	51.3321
liswet12	202	200	−51.3321	−51.3321	−51.3321	−51.3321
liswet2	202	200	34.2425	34.2425	34.2425	34.2425
liswet3	202	200	20.8438	20.8438	20.8438	20.8438
liswet4	202	200	15.1018	15.1018	15.1018	15.1018
liswet5	202	200	323.711	323.711	323.711	323.711
liswet6	202	200	44.3262	44.3262	44.3262	44.3262
liswet7	202	200	50.7545	50.7545	50.7545	50.7545
liswet8	202	200	50.7572	50.7572	50.7572	50.7572
liswet9	202	200	50.7571	50.7571	50.7571	50.7571
lowpass	31	95	22.0009	71.7826	89.6168	41.3995
lsqfit	2	2	0.033787	0.033787	0.033787	0.0337877

TABLE B.8: Results from comparison of the solvers on linear-constraind problems (average of 10 runs for stochastic solvers)- Part 3 .

Name	n	m	f^*	ES-LC-B	ES-LC-P	PSWARM
makela4	21	40	0	0	0	73.0866
Michalewicz1	2	5	-1	-1	-1	-1
nuffield_continuum	2	5	-2.54941	-2.54941	-2.54941	-2.54941
pentagon	6	15	0.000136526	0.000151809	0.000138316	0.000141417
pt	2	201	0.178391	0.178392	0.178391	0.179065
s224	2	6	-304	-304	-304	-304
s231	2	2	$7.36579e-14$	$1.17588e-09$	$1.94008e-12$	0.0244195
s232	2	4	-1	-0.99998	-1	-0.999998
s250	3	7	-3300	-3300	-3300	-3300
s251	3	7	-3456	-3455.99	-3456	-3456
s253	3	4	69.282	69.282	69.282	69.282
s268	5	5	0.173975	0.444779	0.173975	3.04786
s277	4	8	5.07621	5.08267	5.07833	5.08355
s278	6	12	7.84	8.6138	7.85363	7.96406
s279	8	16	10.619	18.0295	10.6574	10.7567
s280	10	20	13.386	48.2646	15.1412	13.7558
s331	2	4	4.25838	4.25838	4.25838	4.25838
s340	3	2	$-1.57675e+66$	$-1.57675e+66$	$-4.79231e+64$	$-2.79811e+15$
s354	4	5	0.113784	0.114795	0.113796	0.114502
s359	5	14	$-5.4702e+06$	$-4.5406e+06$	$-5.26799e+06$	$-5.44339e+06$
s392	30	70	$-1.01391e+06$	-214991	0	-927519
simpllpa	2	4	1	1.00001	1.00285	1
simpllpb	2	5	1.1	1.10001	1.10091	1.1
sipow1	2	200	-1	-1	-1	-1

TABLE B.9: Results from comparison of the solvers on linear-constraind problems (average of 10 runs for stochastic solvers)- Part 4 .

Name	n	m	f^*	ES-LC-B	ES-LC-P	PSWARM
sipow1m	2	200	−1.00012	−1.00012	−1.00012	−1.00012
sipow2	2	101	−1	−0.999999	−1	−1
sipow2m	2	101	−1.00049	−1.00049	−1.00049	−1.00049
sipow3	4	199	0.523141	0.523166	0.523141	0.878149
sipow4	4	200	0.267056	0.267059	0.283533	0.286185
stancmin	3	5	4.25	4.25006	4.25	4.25002
structure2	176	1920	−5.04468	−1.91619	−2.66601	−3.93316
tfi2	3	201	0.649039	0.649051	0.64941	0.661233
weapons	65	77	−1709.63	−1689.01	−1690.43	−1694.94
yao	200	201	27.7202	27.7202	27.7202	27.7202
zecevic2	2	6	−4.125	−4.125	−4.125	−4.125

TABLE B.10: Results from comparison of the solvers on linear-constraind problems (average of 10 runs for stochastic solvers) - Part 5 .

Bibliography

[1] Benchmarks for nonlinear optimization. http://www.princeton.edu/~rvdb/bench.html.

[2] GLOBAL Library. http://www.gamsworld.org/global/globallib.htm.

[3] M. A. Abramson, C. Audet, G. Couture, J. E. Dennis Jr., S. Le Digabel, and C. Tribes. The NOMAD project. Software available at http://www.gerad.ca/nomad.

[4] M. A. Abramson, O. A. Brezhneva, J. E. Dennis Jr., and R. L. Pingel. Pattern search in the presence of degenerate linear constraints. *Optim. Methods Softw.*, 23:297–319, 2008.

[5] G. Aditya, B. Akhilesh, and C. Kuntal. A comprehensive review of image smoothing techniques. *Inter. J. of Advanced Research in Computer Engineering & Technology*, June 2012.

[6] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transfom. *IEEE Trans. Comput.*, 23(1):90–93, Jan. 1974.

[7] M. M. Ali, C. Khompatraporn, and Z. B. Zabinsky. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *J. Global Optim.*, 31:635–672, 2005.

[8] L. Altenberg. Advances in genetic programming. chapter The Evolution of Evolvability in Genetic Programming, pages 47–74. MIT Press, Cambridge, MA, USA, 1994.

[9] P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM J. Matrix Anal. Appl.*, 23(1):15–41, 2001.

[10] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32(2):136–156, 2006.

[11] B. C. Arnold, N. Balakrishnan, and H. N. Nagaraja. *A first course in order statistics (classics in applied mathematics)*. SIAM, 2008.

[12] D. V. Arnold. Optimal weighted recombination. In A. H. Wright, M. D. Vose, K. A. De Jong, and L. M. Schmitt, editors, *Foundations of Genetic Algorithms 8*, volume 3469 of *Lecture Notes in Computer Science*, pages 215–237. Springer-Verlag, Berlin Heidelberg, 2005.

[13] L. Arnold, A. Auger, N. Hansen, and Y. Ollivier. Information-geometric optimization algorithms: A unifying picture via invariance principles. June, 2013.

[14] C. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artif. Intell. Rev.*, 11(1-5):11–73, Feb. 1997.

[15] C. Audet. A short proof on the cardinality of maximal positive bases. *Opti. Lett.*, (5):191–194, 2011.

[16] C. Audet, S. Le Digabel, and C. Tribes. NOMAD user guide. Technical Report G-2009-37, Les cahiers du GERAD, 2009.

[17] C. Audet and J. E. Dennis Jr. Analysis of generalized pattern searches. *SIAM J. Optim.*, 13:889–903, 2002.

[18] C. Audet and J. E. Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM J. Optim.*, 17:188–217, 2006.

[19] C. Audet and J. E. Dennis Jr. A progressive barrier for derivative-free nonlinear programming. *SIAM J. Optim.*, 20(1):445–472, 2009.

[20] A. Auger. Convergence results for the (1,λ)-SA-ES using the theory of ϕ-irreducible Markov chains. *Theor. Comput. Sci.*, 334:35–69, 2005.

[21] A. Auger, D. Brockhoff, and N. Hansen. Benchmarking the local metamodel CMA-ES on the noiseless BBOB'2013 test bed. In *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '13 Companion, pages 1225–1232, New York, NY, USA, 2013. ACM.

[22] A. Auger and N. Hansen. A restart cma evolution strategy with increasing population size. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pages 1769–1776, 2005.

[23] A. Auger, N. Hansen, Z. J. Perez, R. Ros, and M. Schoenauer. Experimental comparisons of derivative free optimization algorithms. In *8th International Symposium on Experimental Algorithms*, number 5526, pages 3–15. Springer Verlag, 2009.

[24] T. Bäck. *Evolutionary algorithms in theory and practice: Evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, Oxford, UK, 1996.

[25] T. Bäck and M. Schütz. Evolution strategies for mixed–integer optimization of optical multilayer systems. In *EVOLUTIONARY PROGRAMMING IV – PROC. FOURTH ANNUAL CONF. EVOLUTIONARY PROGRAMMING (EP-95*, pages 33–51. The MIT Press, 1995.

[26] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evol. Comput.*, 1(1):1–23, Mar. 1993.

[27] A. S. Bandeira, K. Scheinbeerg, and L. N. Vicente. Convergence of trust-region methods based on probabilistic models. Technical report, University of Coimbra, 2013.

[28] A. S. Bandeira, K. Scheinberg, and L. N. Vicente. Computation of sparse low degree interpolating polynomials and their application to derivative-free optimization. *Math. Program.*, 134(1):223–257, 2012.

[29] J.-P. Bérenger. A perfectly matched layer for the absorption of electromagnetic waves. *Journal of Computational Physics*, 114(2):185–200, 1994.

[30] H.-G. Beyer. *The Theory of Evolution Strategies*. Springer, 1998.

[31] H.-G. Beyer. Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. In *Computer Methods in Applied Mechanics and Engineering*, pages 239–267, 2000.

[32] H.-G. Beyer and H.-P. Schwefel. Evolution strategies: A comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

[33] A. Bienvenüe and O. François. Global convergence for evolution strategies in spherical problems: Some simple proofs and difficulties. *Theor. Comput. Sci.*, 306(1-3):269–289, Sept. 2003.

[34] F. Billette, S. L. Bégat, P. Podvin, and G. Lambaré. Practical aspects and applications of 2d stereotomography. *Geophysics*, 68(3):1008–1021, 2003.

[35] F. Billette and G. Lambaré. Velocity macro-model estimation from seismic reflection data by stereotomography. *Geophys. J. Inter.*, 135(2):671–690, 1998.

[36] A. J. Booker, J. E. Dennis Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural and Multidisciplinary Optimization*, 17:1–13, 1998.

[37] Z. Bouzarkouna. *Well placement optimization*. PhD thesis, University Paris-Sud - Laboratoire de Recherche en Informatique, 2012.

[38] R. Brossier. *Imagerie sismique à deux dimensions des milieux visco-élastiques par inversion des formes d'ondes : développements méthodologiques et applications*. PhD thesis, Geoazur, Sophia Antipolis - UFR Sciences, 2009.

[39] L. Bull. Learning classifier systems: A brief introduction. In *In Bull, L (Ed.): Applications of Learning Classifier Systems. Berlin u.a*, page 14. Springer, 2004.

[40] H. Calandra, S. Gratton, R. Lago, X. Pinel, and X. Vasseur. Two-level preconditioned krylov subspace methods for the solution of three-dimensional heterogeneous helmholtz problems in seismics. *Numerical Analysis and Applications*, 5(2):175–191, mai 2012.

[41] L. M. Carvalho, S. Gratton, R. Lago, and X. Vasseur. A flexible generalized conjugate residual method with inner orthogonalization and deflated restarting. *SIAM J. Matrix Anal. Appl.*, 32(4):1212–1235, Nov. 2011.

[42] R. Chiong, T. Weise, and Z. Michalewicz, editors. *Variants of evolutionary algorithms for real-world applications*. Springer-Verlag: Berlin/Heidelberg, 2011.

[43] F. H. Clarke. *Optimization and nonsmooth analysis*. John Wiley & Sons, New York, 1983. Reissued by SIAM, Philadelphia, 1990.

[44] C. A. C. Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191:1245–1287, 2002.

[45] C. A. C. Coello and E. M. Montes. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 16:193–203, 2002.

[46] G. Cohen. *Higher-order numerical methods for transient wave equations*. Springer-Verlag, Berlin, Germany, 2002.

[47] M. D. Collins and W. A. Kuperman. Nonlinear inversion for ocean bottom properties. *The Journal of the Acoustical Society of America*, (92):2770–2782, 1992.

[48] A. R. Conn and S. L. Digabel. Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. *Optim. Methods Softw.*, 28(1):139–158, 2013.

[49] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-region Methods*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, PA, USA, 2000.

[50] A. R. Conn, K. Scheinberg, and L. N. Vicente. Geometry of sample sets in derivative-free optimization: polynomial regression and underdetermined interpolation. *Math. Program.*, 28(4):721–748, 2008.

[51] A. R. Conn, K. Scheinberg, and L. N. Vicente. Global convergence of general derivative-free trust-region algorithms to first and second order critical points. *SIAM J. Optim.*, 20:387–415, 2009.

[52] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to derivative-free optimization*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, 2009.

[53] A. L. Custódio, H. Rocha, and L. N. Vicente. Incorporating minimum Frobenius norm models in direct search. *Comput. Optim. Appl.*, 46:265–278, 2010.

[54] A. L. Custódio and L. N. Vicente. Using sampling and simplex derivatives in pattern search methods. *SIAM J. Optim.*, 18(2):537–555, 2007.

[55] I. Daubechies. *Ten lectures on wavelets*. MPS-SIAM. SIAM, Philadelphia PA, 1992.

[56] C. Davis. Theory of positive linear dependence. *Amer. J. Math.*, 76(4):733–746, 1954.

[57] Y. Diouane, H. Calandra, S. Gratton, and X. Vasseur. A parallel evolution strategy for acoustic full-waveform inversion: Extended abstract. In *EAGE High Performance Computing for Upstream Workshop in Chania, Crete, Greece*, 2014.

[58] Y. Diouane, S. Gratton, and L. N. Vicente. Globally convergente evolution strategies. *Math. Program.*, to appear. doi: 10.1007/s10107-014-0793-x.

[59] Y. Diouane, S. Gratton, and L. N. Vicente. Globally convergent evolution strategies for constrained optimization. Technical report, CERFACS, Toulouse, France, TR-PA-14-50.

[60] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91:201–213, 2002.

[61] E. D. Dolan, J. J. Moré, and T. S. Munson. Optimality measures for performance profiles. *SIAM J. Optim.*, 16:891–909, 2006.

[62] G. Fasano, J. L. Morales, and J. Nocedal. On the geometry phase in model-based algorithms for derivative-free optimization. *Optim. Methods Softw.*, 24(1):145–154, 2009.

[63] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Math. Program.*, 91:239–269, 2002.

[64] D. B. Fogel. *System identification through simulated evolution: A machine learning approach to modeling*. Ginn Press, 1991.

[65] D. B. Fogel. *Evolving artificial intelligence*. PhD thesis, La Jolla, CA, USA, 1992.

[66] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3:1–16, 1995.

[67] S. Forrest and A. S. Perelson. Genetic algorithms and the immune system. In H.-P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature*, volume 496 of *Lecture Notes in Computer Science*, pages 319–325. Springer, 1991.

[68] O. Gauthier, J. Virieux, and A. Tarantola. Two dimensional nonlinear inversion of seismic waveforms: Numerical results. *Geophysics*, 51(7):1387–1403, July 1986.

[69] S. Gelly, S. Ruette, and O. Teytaud. Comparison-based algorithms are robust and randomized algorithms are anytime. *Evol. Comput.*, 15(4):411–434, Dec. 2007.

[70] P. Gerstoft. Nonlinear inversion for ocean bottom properties. *The Journal of the Acoustical Society of America*, 95(2):770–782, 1994.

[71] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr, a Constrained and Unconstrained Testing Environment, revisited. *ACM Trans. Math. Software*, 29:373–394, 2003.

[72] S. Gratton, Ph. L. Toint, and A. Tröltzsch. An active-set trust-region method for derivative-free nonlinear bound-constrained optimization. *Optim. Methods Softw.*, 26:873–894, 2011.

[73] S. Gratton, C. W. Royer, L. N. Vicente, and Z. Zhang. Direct search based on probabilistic descent. Technical report, University of Coimbra, 2014.

[74] S. Gratton and L. N. Vicente. A merit function approach for direct search. Technical report, 2014.

[75] G. W. Greenwood and Q. J. Zhu. Convergence in evolutionary programs with self-adaptation. *Evolutionary Computation*, 9:57–147, 2001.

[76] B. Greer. Numerical optimization with neuroevolution. In *In Proceedings of the 2002 Congress on Evolutionary Computation (CEC2002*, 2002.

[77] J. D. Griffin, T. G. Kolda, and R. M. Lewis. Asynchronous parallel generating set search for linearly-constrained optimization. *SIAM J. Sci. Comput.*, 30:1892–1924, 2008.

[78] N. Hansen. The CMA Evolution Strategy: A tutorial. June 28, 2011.

[79] N. Hansen, A. Auger, R. R. Raymond, S. Finck, and P. Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking bbob-2009. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '10, pages 1689–1696, New York, NY, USA, 2010. ACM.

[80] N. Hansen, S. Fincky, R. Rosz, and A. Auger. Real-parameter black-box optimization benchmarking 2010: Noisy functions definitions. Technical report, March 22, 2010.

[81] N. Hansen, S. Fincky, R. Rosz, and A. Auger. Real-parameter black-box optimization benchmarking 2010: Noiseless functions definitions. Technical report, September 28, 2010.

[82] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evol. Comput.*, 11(1):1–18, Mar. 2003.

[83] N. Hansen, A. S. P. Niederberger, L. Guzzella, and P. Koumoutsakos. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Trans. Evolutionary Computation*, 13:180–197, 2009.

[84] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 312–317, 1996.

[85] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.*, 9(2):159–195, June 2001.

[86] N. Hansen, A. Ostermeier, and A. Gawelczyk. On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. In L. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms, Pittsburgh*, pages 57–64, 1995.

[87] A. Hedar and M. Fukushima. Derivative-free filter simulated annealing method for constrained continuous global optimization. *J. Global Optim.*, 35:2006, 2004.

[88] A. Henderson. *ParaView Guide.* A Parallel Visualization Application. Kitware Inc, 2007.

[89] F. Herrera, M. Lozano, and J. L. Verdegay. Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artif. Intell. Rev.*, 12(4):265–319, Aug. 1998.

[90] W. Hock and K. Schittkowski. *Test Examples for Nonlinear Programming Codes.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1981.

[91] J. H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[92] J. H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control and artificial intelligence.* MIT Press, Cambridge, MA, USA, 1992.

[93] W. Huyer and A. Neumaier. Global optimization by multilevel coordinate search. *J. Global Optim.*, 14:331–355, 1999.

[94] L. Ingber. Adaptive simulated annealing (ASA). *Global Optimization C-code*, 1993.

[95] L. Ingber and B. Rosen. Genetic algorithms and very fast simulated reannealing: A comparison. *Math. Comput. Modelling*, 16:87–100, 1992.

[96] J. Jägersküpper. How the (1+1)-ES using isotropic mutations minimizes positive definite quadratic forms. *Theor. Comput. Sci.*, 361:38–56, 2006.

[97] J. Jägersküpper. Probabilistic runtime analysis of (1+1)-ES using isotropic mutations. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, GECCO '06, pages 461–468, New York, NY, USA, 2006. ACM.

[98] J. Jahn. *Introduction to the Theory of Nonlinear Optimization.* Springer-Verlag, Berlin, 1996.

[99] E. T. Jaynes. Where do we stand on maximum entropy? In *Maximum Entropy Formalism Conference.* Massachusetts Institute of Technology, Apr. 1978.

[100] M. Jebalia and A. Auger. Log-linear convergence of the scale-invariant (μ/μ_w,λ)-ES and optimal μ for intermediate recombination for large population sizes. In *PPSN (1)*, pages 52–62, 2010.

[101] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft. Comput.*, 9(1):3–12, Jan. 2005.

[102] C. T. Kelley. *Implicit filtering.* Number 23 in Software Environments and Tools. SIAM, Philadelphia, PA, USA, 2011.

[103] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.

[104] B. L. Kennett, M. S. Sambridge, and P. R. Williamson. Subspace methods for large inverse problems with multiple parameter classes. *Geophys. J. Int.*, 94:237–247, 1988.

[105] S. Kern, N. Hansen, and P. Koumoutsakos. Local meta-models for optimization using evolution strategies. In *Parallel Problem Solving from Nature - PPSN IX*, pages 939–948. Springer, 2006.

[106] S. Kim and D. Zhang. Convergence properties of direct search methods for stochastic optimization. In *Winter Simulation Conference*, pages 1003–1011. WSC, 2010.

[107] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[108] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Rev.*, 45:385–482, 2003.

[109] T. G. Kolda, R. M. Lewis, and V. Torczon. A generating set direct search augmented lagrangian algorithm for optimization with a combination of general and linear constraints. Technical report, Sandia National Laboratories, USA, 2006.

[110] T. G. Kolda, R. M. Lewis, and V. Torczon. Stationarity results for generating set search for linearly constrained optimization. *SIAM J. Optim.*, 17:943–968, 2006.

[111] S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evol. Comput.*, 7(1):19–44, Mar. 1999.

[112] O. Kramer. A review of constraint-handling techniques for evolution strategies. *Applied Computational Intelligence and Soft Computing*, 2010:1–11, 2010.

[113] R. Lago. *A study on block flexible iterative solvers with application to Earth imaging problem in geophysics*. PhD thesis, Institut National Polytechnique de Toulouse, 2013.

[114] G. Lambaré. Stereotomography. *Geophysics*, 73(5):VE25–VE34, 2008.

[115] J. Larson and S. C. Billups. Stochastic derivative-free optimization using a trust region framework. Technical report, 2014.

[116] S. Le Digabel. Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. *toms*, 37:1–15, 2011.

[117] R. M. Lewis and V. Torczon. Pattern search methods for linearly constrained minimization. *SIAM J. Optim.*, 10:917–941, 2000.

[118] R. M. Lewis and V. Torczon. A globally convergent augmented lagrangian pattern search algorithm for optimization with general constraints and simple bounds. *SIAM J. Optim.*, 12(4):1075–1089, 2002.

[119] M. Locatelli. A note on the Griewank test function. *J. Global Optim.*, 25:169–174, 2003.

[120] S. Lucidi, M. Sciandrone, and P. Tseng. Objective-derivative-free methods for constrained optimization. *Math. Program.*, 92:37–59, 1999.

[121] J. Mathorel. Implémentation d'un algorithme sans gradient de résolution du problème inverse en sismique: Master thesis. Technical report, 2013.

[122] J. Matyas. Random optimization. *Automation and remote control*, 26:244–251, 1965.

[123] Z. Michalewicz and M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evol. Comput.*, 4:1–32, 1996.

[124] M. Mongeau, H. Karsenty, V. Rouzé, and J.-B. Hiriart-Urruty. Comparison of public-domain software for black box global optimization. *Optim. Methods Softw.*, 13:203–226, 2000.

[125] J. J. Moré and S. M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM J. Optim.*, 20:172–191, 2009.

[126] W. A. Mulder and R. E. Plessix. Exploring some issues in acoustic full waveform inversion. *Geophysical Prospecting*, 56(6):827–841, Nov. 2008.

[127] O. M. Nabi and L. Xiaodong. A comparative study of CMA-ES on large scale global optimisation. In J. Li, editor, *Australasian Conference on Artificial Intelligence*, volume 6464 of *Lecture Notes in Computer Science*, pages 303–312. Springer, 2010.

[128] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[129] Y. Nesterov. Random gradient-free minimization of convex functions. Technical report, Feb. 2011.

[130] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer series in operations research and financial engineering. Springer, New York, NY, 2. ed. edition, 2006.

[131] G. Nolet. *Seismic Tomography: With Applications in Global Seismology and Exploration Geophysics*. D. Reidel publishing Company, 1987.

[132] D. W. Oldenburg, P. R. McGillivray, and R. G. Ellis. Generalized subspace methods for large-scale inverse problems. *Geophys. J. Int.*, 114:12–20, 1993.

[133] S. Operto, J. Virieux, P. Amestoy, J.-Y. L'Excellent, L. Giraud, and H. Ben-Hadj-Ali. 3d finite-difference frequency-domain modeling of visco-acoustic wave propagation using a massively parallel direct solver: A feasibility study. *Geophysics*, 72(5):SM195–SM211, Sept. 2007.

[134] S. Operto, J. Virieux, J. X. Dessa, and G. Pascal. Crustal seismic imaging from multifold ocean bottom seismometer data by frequency domain full waveform tomography: Application to the eastern nankai trough. *J. Geophys. Res.*, 159(3):1032–1056, 2006.

[135] X. Pinel. *A perturbed two-level preconditioner for the solution of three-dimensional heterogeneous Helmholtz problems with applications to Geophysics*. PhD thesis, CERFACS and INPT, 2010.

[136] R. E. Plessix. A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. *Geophys. J. Inter.*, 167(2):495–503, Nov. 2006.

[137] M. J. D. Powell. Least Frobenius norm updating of quadratic models that satisfy interpolation conditions. *Math. Program.*, 100(1):183–215, 2004.

[138] M. J. D. Powell. The newuoa software for unconstrained optimization with derivatives. Technical report, University of Cambridge, UK, 2004.

[139] G. R. Pratt and M. H. Worthington. Inverse theory applied to multi-Source cross-hole tomography. Part 1: acoustic wave-equation method. *Geophysical Prospecting*, 38(3):287–310, Apr. 1990.

[140] R. G. Pratt. Seismic waveform inversion in the frequency domain, part 1: Theory and verification in a physical scale model. *Geophysics*, 64:888–901, 1999.

[141] C. Ravaut, S. Operto, L. Improta, J. Virieux, A. Herrero, and P. Dell'Aversana. Multiscale imaging of complex structures from multifold wide-aperture seismic data by frequency-domain full-waveform tomography: application to a thrust belt. *Geophys. J. Inter.*, 159(3):1032–1056, 2004.

[142] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution*. Frommann-Holzboog, 1973.

[143] R. G. Reynolds, Z. Michalewicz, and M. J. Cavaretta. Using cultural algorithms for constraint handling in GENOCOP. In *Evolutionary Programming*, pages 289–305, 1995.

[144] J. T. Richardson, M. R. Palmer, G. E. Liepins, and M. Hilliard. Some guidelines for genetic algorithms with penalty functions. In *Proceedings of the third international conference on Genetic algorithms*, pages 191–197, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.

[145] L. Rios and N. Sahinidis. Derivative-free optimization: A review of algorithms and compariason of software implementations. *J. Global Optim.*, 56:1247–1293, 2013.

[146] M. Robbé and M. Sadkane. Exact and inexact breakdowns in the block gmres method. *Linear Algebra and its Applications*, 419(1):265–285, 2006.

[147] T. P. Runarsson. Constrained evolutionary optimization by approximate ranking and surrogate models. In X. Yao, E. K. Burke, J. A. Lozano, J. Smith, J. J. M. Guervós, J. A. Bullinaria, J. E. Rowe, P. Tino, A. Kabán, and H.-P. Schwefel, editors, *PPSN*, volume 3242 of *Lecture Notes in Computer Science*, pages 401–410. Springer, 2004.

[148] Y. Saad. *Iterative Methods for Sparse Linear Systems, Second Edition*. Society for Industrial and Applied Mathematics, 2 edition, Apr. 2003.

[149] K. Scheinberg and Ph. L. Toint. Self-correcting geometry in model-based algorithms for derivative-free unconstrained optimization. *SIAM J. Optim.*, 20(6):3512–3532, 2010.

[150] H.-P. Schwefel. *Evolutionsstrategie und numerische optimierung*. PhD thesis, 1975.

[151] H. P. Schwefel. *Evolution and optimum seeking: The sixth generation*. John Wiley & Sons, Inc., New York, NY, USA, 1993.

[152] C. Shin and Y. H. Cha. Waveform inversion in the Laplace domain. *Geophys. J. Inter.*, 173:922–931, 2008.

[153] C. Shin and W. Ha. A comparison between the behavior of objective functions for waveform inversion in the frequency and Laplace domains. *Geophysics*, 73(5):VE119–VE133, 2008.

[154] C. Shin and W. Ha. Laplace-domain full-waveform inversion of seismic data lacking low-frequency information. *Geophysics*, 77(5):R199–R206, 2012.

[155] L. Sirgue. The importance of low frequency and large offset in waveform inversion. In *Extended Abstracts*, 68th Confererence & Technical Exhibition. EAGE, 2006.

[156] L. Sirgue and R. G. Pratt. Efficient waveform inversion and imaging : A strategy for selecting temporal frequencies. *Geophysics*, 69.

[157] J. Skilling and R. K. Bryan. Maximum entropy image reconstruction - general algorithm. *Monthly Notices of the Royal Astronomical Society*, 211(1):111–124, 1984.

[158] J. C. Spall. *Introduction to stochastic search and optimization: Estimation, simulation, and control*. John Wiley and Sons, 2003.

[159] R. Storn and K. Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, Dec. 1997.

[160] G. Strang. The discrete cosine transform. *SIAM Review*, 41:135–147, 1999.

[161] A. Tarantola. *Inverse problem theory and methods for model parameter estimation*. Siam, 2005.

[162] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision*.

[163] V. Torczon. On the convergence of pattern search algorithms. *SIAM J. Optim.*, 7(1):1–25, Jan. 1997.

[164] A. I. F. Vaz and L. N. Vicente. A particle swarm pattern search method for bound constrained global optimization. *J. Global Optim.*, 39:197–219, 2007.

[165] A. I. F. Vaz and L. N. Vicente. Pswarm: A hybrid solver for linearly constrained global derivative-free optimization. *Optim. Methods Softw.*, 24:669–685, 2009.

[166] L. N. Vicente and A. L. Custódio. Analysis of direct searches for discontinuous functions. *Math. Program.*, 133:299–325, 2012.

[167] J. Virieux and S. Operto. An overview of full-waveform inversion in exploration geophysics. 74(6):WCC1–WCC26, Nov. 2009.

[168] S. M. Wild and C. A. Shoemaker. Global convergence of radial basis function trust-region algorithms for derivative-free optimization. *SIAM Review*, 55(2):349–371, 2013.

[169] G. Yin, G. Rudolph, and H. P. Schwefel. Analyzing the (1, λ) evolution strategy via stochastic approximation methods. *Evol. Comput.*, 3(4):473–489, Dec. 1995.

[170] Y. Zhang and L. Gao. On numerical solution of the maximum volume ellipsoid problem. *SIAM J. Optim.*, 14:53–76, 2003.